Faculdade de Engenharia da Universidade do Porto

Low-cost device for live cell imaging applications

José Manuel Salazar Ribeiro

Supervisor at FEUP: João Paulo Cunha
Co-supervisor at I3S: Paula Sampaio

Integrated Master in Bioengineering

September 2016

Faculdade de Engenharia da Universidade do Porto

Low-cost device for live cell imaging applications

José Manuel Salazar Ribeiro

Dissertation submitted to Faculdade de Engenharia da Universidade do Porto to obtain the degree of

Master in Bioengineering

September, 2016

Resumo

O estudo de processos dinâmicos a nível celular é fundamental para o conhecimento da biologia celular e uma das tecnologias utilizadas para essa análise é a microscopia de células vivas. Em certos tipos de experiências, são necessários sistemas de perfusão de fluídos, de modo a complementarem e assegurarem todos os requisitos necessários para a sua realização, como a troca de meio de cultura e a adição de diferentes compostos. Os parâmetros de funcionamento pretendidos pelo operador da experiência devem ser inseridos e estes sistemas devem executar, de forma automática, o que lhe foi especificado.

No entanto, muitas vezes, estes sistemas de perfusão de fluídos podem ser extremamente dispendiosos e o conceito *open-source* tem sido visto como uma solução plausível para ultrapassar esta barreira financeira. Esta abordagem tem sido utilizada em vários ambientes, inclusive para a produção de equipamento laboratorial. Alguns grupos de investigação têm mesmo divulgados os equipamentos produzidos pelos mesmos, motivados pela redução de custos. Este conceito foi também usado para a criação de um *software* de microscopia (*Micro-Manager*), de modo a integrar vários equipamentos e com elevado potencial de flexibilidade.

Desta forma, este projeto surge com o objetivo de produzir um sistema de entrega controlada de fluidos numa câmara de cultura celular de baixo custo. Foi desenvolvido um sistema constituído por três seringa motorizadas *open-source* que foram adaptadas para aplicações de microscopia de células vivas.

De seguida, o sistema foi integrado no *software* de microscopia *Micro-Manager*. Estas bombas podem ser controladas tanto através do seu ambiente de *script* como também a partir da sua ferramenta de aquisição de imagens multidimensional. A comunicação entre o *software* e as bombas de seringa é intermediada por uma placa de microcontrolador *Arduino*.

A solução final foi testado com duas provas de conceito: troca de meio de cultura e infusão de drogas no meio de cultura num instante específico de tempo. No primeiro foram utilizados fibroblastos incubados com uma droga que inibia a mitose destas mesmas células. Tal como esperado, quando realizada a substituição do meio de cultura por um meio sem droga, as células retomaram a sua normal atividade biológica sendo possível verificar a mitose destas mesmas células. Já no segundo teste foi introduzido um marcador de fluorescência que permitiu observar a localização das mitocôndrias no interior da célula.

O trabalho realizado possibilitou produzir uma solução de baixo custo comparativamente com outras soluções disponíveis no mercado. O sistema final foi testado para seringas entre 1 e 50 mL tendo sido obtidas uma exatidão menor que 2%, reprodutibilidade menor 1% e resolução máxima de aproximadamente 1 nL. No entanto, este projeto ainda se encontra numa fase inicial, pelo que muitos melhoramentos podem vir ser a realizados.

Abstract

The study of cell dynamic processes is fundamental for the knowledge of cell biology and live cell imaging is one of the approaches used to do it. For some kind of experiments, dynamic change of fluids in culture are required in order to complement and ensure all need criteria for its performance as medium change and compound addiction. The user must insert the desired functioning parameters and the system should be able to execute what was specified automatically.

However, commonly these fluid perfusion systems are very expensive and the open-source concept seems to be a good solution for overcome this problem. It is already being used in several environments, including at laboratory equipment development. There are already some research groups that shared how they produced their own equipment in order to reduce costs. It was also developed an open-source microscopy software (Micro-Manager) in order to integrate different microscopy hardware with high flexibility potential.

Thus, this project aims to develop a low cost fluid delivery system in a controlled way to a cell culture chamber. The system has three open source syringe pumps which were adapted and build to fulfil live cell imaging applications.

Then, the system was integrated with the microscopy software Micro-Manager. These pumps can be controlled not only in its script environment but also through its multi-dimensional image acquisition tool.

The final system was tested in two proof of concept experiments: medium replacement and drug infusion at a specific time instance. In the first experiment, fibroblasts were incubated with a specific drug, which inhibit cell mitosis. As expected, when performing culture medium replacement with fresh medium without the mentioned drug, these cells resumed their normal biological activity and it could be observed their mitosis. Lastly, it was simulated cell culture infusion with a drug at a specific time instance. In the second experiment, a fluorescence marker was infused and it was observed the mitochondria arrangement within the cell.

It was build a solution that is low cost compared with commercial ones. The final system was tested for syringes between 1 and 50 mL and it was obtained accuracy less than 2%, reproducibility less than 1% and a maximum resolution around 1 nL. However, this project is at an initial stage so, a lot of improvements can be performed in order to achieve a better and suitable solution.

Acknowledgments

I would like to thank to:

Professor João Paulo Cunha, my supervisor at FEUP, for all the support, discussions, patience and guidance.

Professor Paula Sampaio, my supervisor at I3S, for all the dedication, suggestions, commitment and motivation.

BRAIN research group, for all the knowledge exchange and introducing me to research experience.

I3S researchers, for providing a good research environment and motivating me with lot of experiments suggestions for this project. A special acknowledgement to Joana Macedo and Rui Ribeiro, who affably prepared the cells used on proof of concept experiments.

All my friends, for all laughs proportionated and being always there for me, namely Maria Magalhães for all the support especially when a little incentive was needed.

Finally, my mother, father, grandparents and Beatriz for their unconditional support.

José Ribeiro

Table of Contents

List of figures

List of tables

Abbreviations

ADK	Assessment and Deployment Kit
AOTF	Acousto-Optic Tunable Filter
BRET	Bioluminescence Resonance Energy Transfer CO2 – Carbon dioxide
DAC	Digital-to Analog Converter
DC motor	Direct current motor
CAD	Computer Aided Design
CAN	Controller Area Network
COM	Communication port
CO_2	Carbon dioxide
CPU	central processing unit
DIY	Do It Yourself
DLL	Dynamic-link library
DNA	Deoxyribonucleic Acid
EDL	Electric Double Layer
EEPROM	Electrically-Erasable Programmable Read-Only Memory
FRET	Forster Resonance Energy Transfer
FTDI	Future Technology Devices International
GPIO	General Programmable Input/Output
GPU	Graphics Processing Unit
GUI	Graphical user interface
HD	High Definition
HDMI	High-Definition Multimedia Interface
HID	Human Interface Device
ID	inner diameter
IDE	Integrated Development Environment
IR	Infrared
ISP	Internet Service Provider
KB	Kilobyte
LCD	Liquid-crystal display

LED	Light Emitting Diode
MDA	Multi-Dimensional Acquisition
MOST	Michigan Tech in Open Sustainability Technology
NEMA	National Electrical Manufacturers Association
OD	Outer diameter
OS	Operating System
PDMS	Polydimethylsiloxane
PLA	Polylactic acid
PWM	Pulse-Width Modulation
RNAi	Interference Ribonucleic Acid
Sd	Standard Deviation
SD	Secure Digital
SLA	Stereolithography
SLI	SLIcing
SLS	Selective Laser Sintering
SRAM	Static Random Access Memory
SoC	System on a Chip
STL	STereoLithgraphy
STLC	S -Trityl-L-cysteine
TI	Texas Instruments
WPI	World Precision Instruments

Chapter 1

Introduction

1.1. Motivation

Cell processes are highly dynamic and it is essential to use techniques of live cell imaging, which allow real-time cells and subcellular structures video recording, to study their behaviour. Current techniques are based on the use of time-lapsed microscopes, which allow capturing an image sequence of living cells, and visualising their behaviour when subjected to different stimuli. For that purpose, automated delivery micro and nano-scale fluids products and a good synchronization control mechanism, with an image acquisition system and control of the motorized stage, are necessary. This is a really difficult step because these techniques are done at micro to nano scale.

There are several products in the market that are specific for this desired purpose. However they are very expensive and not open to modifications in order to adapt to new applications. Therefore, it is required to develop affordable and upgradable devices to perform these kinds of experiments, with specific and demanding perfusion and compounds addition tasks, trigger times and coordination.

1.2. Objectives

Taking into account the referred problem, the main purpose of this project is the development of a low-cost device that can produce dynamic changes in living cell medium, as simple culture medium replacement or adding drugs or nanomaterials, at specific times during image acquisition, in order to evaluate resulting cell behaviour. At same time, it should be practical and with the same characteristics as the other existent solutions. More specifically, it is supposed to create an automated system that allows controlled exchange of fluids with a cell culture chamber.

At a later stage, the system created should be integrated in the specific open-source microscopy software (Micro-manager[1]). Its source code should be adapted and enhanced to perform image acquisition with the automated fluid delivery solution mentioned above, in a synchronized way.

1.3. Contributions

The proposed work might contribute in several ways for the technology and research environment, including:

- Design and development of a low-cost system, capable of infusing and withdrawing fluids to a cell culture chamber;
- Control the above mentioned system through the open-source microscopy software Micro-Manager, using a user-friendly GUI;
- Perform proof of concept experiments, namely to test main cell culture fluid exchange tasks, medium replacement and drug infusion.

1.4. Overview of Dissertation

Besides this Introduction, this dissertation includes four chapters. Chapter 2 reviews some related work already published. It includes also an explanation of the main concepts related with microscopy, live cell imaging, open-source paradigm, microcontrollers and 3D-printing. Further, it includes a microscopy fluid delivery technology survey. In Chapter 3, it is described the research methodology approach, and its results are presented and discussed in Chapter 4. Lastly, Chapter 5 presents the major conclusions of this dissertation, with main future work suggestions.

Chapter 2

Literature Review

2.1. Microscopy

Microscopy is the technical field that makes use of an instrument, the microscope to view objects and details that cannot be observed with naked eye. The microscope must produce a magnified image, resolve the details and make them visible to the eye, camera or imaging device. Microscopes can be classified based on the physical principle that is used to generate an image: optical if they use visible light, X-ray if they use a beam of x-rays, scanning acoustic if they use sound waves, or electron if they use a beam of high energy electrons. [2][3]

The light microscopy is probably the most well-known and used in the research community. These microscopes use optical components and visible light to generate in images by several different techniques that could be aggregated two groups: transmission microscopy that includes bright-field, phase contrast or differential interference contrast; and fluorescence microscopy, which includes a broad range of techniques such as wide-field, laser scanning confocal and two-photon microscopy. Also the new “super-resolution” techniques are base in fluorescence microscopy.

Phase contrast microscopy exploits the fact that light slows slightly when passing through biological specimens. The specimen is illuminated by a hollow cone of light coming through a phase annulus in the condenser. Light rays passing through the specimen are deviated and slightly retarded, whose amplitude and phase change with the medium properties, which are detected by a phase plate. The phase shifts are converted in brightness variations. This is useful to reveal unstained cellular structures that are not visible with a simple bright-field microscopy and allows the visualization of live cells.

In the fluorescence microscopy, the application of an array of fluorochromes make possible to identify cells and sub-microscopic cellular components with a high degree of specificity. The fluorescence microscope is capable of revealing the presence of single molecules and to identify simultaneously different molecules by targeting with multiple fluorescence labels. In a wide-field fluorescence microscope, the light beam passes through excitation filter to isolate a specific range. Then it is reflect by a dichroic beam splitter to the objective lens and illuminate the specimen. The filtered light beam is capable to selectively excite the electrons in fluorochrome molecules that are present in the object. When excited electrons return to their ground state emits light with higher wavelength. The emitted light collected by the objective passes thought dichroic beam splitter to separate the much weaker emitted fluorescence from the excitation light and is filtered by the emission filter to avoid

crosstalk with other fluorochromes. The signal is detectable through the oculars of the microscope, or registered by a digital camera. The image is generated continuously, across the entire field of view.

Laser scanning confocal microscopy is a fluorescence technique that uses point illumination and a pinhole in an optically conjugate plane in front of the detector to eliminate out-of-focus signal. As only light produced by fluorescence very close to the focal plane can be detected, the image's optical resolution, particularly in the sample depth direction, is much better than that of wide-field microscopes. This occurs because the detection pinhole does not permit rays of light from out-of-focus points to pass through it. To obtain a full image, the point of light is moved across the specimen by scanning mirrors. The emitted/reflected light passing through the detector pinhole is transformed into electrical signals by a photomultiplier and usually displayed on a computer monitor [9][10].

2.1.1. Live cell imaging

Nowadays life science research is increasingly focusing on studying dynamic processes like cell migration, morphological changes of cells, organs or whole animals and physiological such as changes of intracellular ion composition. One approach to address these challenging demands in living specimens is to image live cells with microscope systems adapted for the job. Live-cell imaging allows investigation of dynamical processes of living cells instead of giving a “snapshot” of a cell’s current state. It provides spatial and temporal information of dynamic events in single cells, cellular networks (in situ) or even whole organisms (in vivo). These features make live-cell imaging a valuable technique for addressing questions in cell biology, cancer research, developmental biology and neuroscience [2][3].

Imaging of live cells use time-lapse microscopy in order to capture behaviour changes and conclude about their response when these cells are exposed to different stimulus.

There are important aspects that should be take into account when performing live cell imaging protocols because cells are very sensitive to UV light, IR light and even fluorescence excitation. So, efforts should be made to reduce the cells exposition time to light in order to minimize phototoxicity and maintain cell viability. Another aspects to be aware are the importance of avoiding cell stress, which can seriously affect cell viability and proliferation. Some crucial factors are the culture medium and its components, temperature, pH and CO_2, and osmolarity. To maintain temperature at suitable levels, there is a large box that encloses the entire microscope or stage-top incubators which only encloses the recipient where the cells are incubated.[6] Furthermore, suitable and proper fluorescent proteins and highlighters should be used in order to signalize the pretended structures or molecules and follow theirs dynamics.

The cells could be cultured as static cultures or as perfusion cultures. Static cultures were the first to be used and they are still the most commonly used cell culture technique. Basically, the cells are cultivated in a proper recipient (Petri dish or multiwall plate) that contains culture medium, which is added or removed with necessarily human operated devices (like pipettes). On the other side, in perfusion cultures, the cells are added to a culture chamber equipped with some input/output channels that will allow the inlet/outlet of fluids through automatic fluid deliver systems whose actuation can be defined by the human who operates the system.[7]

For the live cell imaging techniques that require exchange of media and/or addition of compounds, it is preferable to use perfusion cultures because some problems, like contamination risks and medium fluctuations, from continuous manual control of medium replacement methods can be avoided by the use of accurate perfusion control systems. However, perfusion cultures are more complex and expensive systems.

2.1.1.1. Applications

The range of microscopic techniques applied for live-cell imaging is very wide because this is being integrated with specialized techniques for monitoring, measuring, and disturbing dynamic activities of cells and subcellular structures.

Common applications of live cell imaging are to analyse the dynamic progression of cells in specific conditions. In the chemotaxis assays, the cells are exposed to a chemical stimuli and the objective is to check and report their behaviour changes and movement. In the study of angiogenesis process, it is observed the growth of new blood vessels from pre-existing ones, and is very common to stimulate or inhibit this process with chemical stimulation with the addition of growth factor, for the first case, and drugs, in the second ones. Live cell imaging is also used to perform cell migration assays. The mechanisms used by cells to move are mainly driven by flagella or cilia, in prokaryotic, and cytoskeleton in eukaryotic. One specific type of cell migration is the wound healing process where the cells migrate in a specific order to a trauma placed in a body's local. The analysis of cell cycle durations and mitotic process in result of diverse perturbations such as RNAi or addition of active compounds is also an important application of live cell imaging.

Live cell fluorescent imaging is also useful to study the dynamic protein intracellular interactions, metabolic and signalling pathways such as the regulatory pathway between the extracellular matrix and the cell itself. For this, Forster or bioluminescence resonance energy transfer events are observed. These are useful tools that allow the quantification of molecular dynamics that occur in cell domain, namely the protein-protein interactions, mitosis, protein-DNA interactions and even conformational changes suffered by the proteins. In the FRET assay derivatives of green fluorescent proteins, cyan and yellow fluorescent proteins are used and they are attached to the proteins of interest through molecular biology methods. In turn, cyan fluorescent protein fluorescent protein is excited with fluorescent light and transfers the energy to the yellow fluorescent protein when the proteins of interest are close enough (<10 nm). In the BRET case, the energy donor is a bioluminescent molecule but the acceptor is also a fluorescent protein.[8]

Another important application of live cell imaging is the study of how shear stress can influence cell behaviour changes. The cell culture is exposed to continuous/discontinuous culture medium flows that can be controlled by the lab operator. The direction of inlet/outlet channels can be varied, representing a variable parameter. This experiences require mainly a closed culture chamber with some channels to allow the perfusion with controlled flows.[9]

2.1.1.2. Perfusion Systems

As mentioned in the section before, some applications in live cell imaging that requires dynamic change of fluids in culture, mainly culture medium change and drug delivery mechanisms. Both of these examples need a perfusion system that is obtained with a pump system and a proper channel from the recipient holding the fluid to the culture chamber. In the following sections, it will be explained three of the main principle used to achieve this controlled fluid delivery: the pressure induced perfusion, peristaltic pumps and syringe pumps. [7] (Appendix A -Table A 1)

Pressure induced perfusion

In this technique, a closed reservoir has two holes: one for inlet and one for outlet. The inlet is connected to a compressor and the outlet is connected to the culture chamber. The pressurized air will enter the reservoir and produce a tension that should be released

through the outlet hole. This way, and soaking the output channel into the fluid contained in the reservoir, that tension would be released by sending out an amount of fluid. It is regularly called a pressure-driven flow since and the output flux of fluid is proportional to the input pressure. The main advantages of this technique are the possibility to create a pulseless flow leading to a very stable system.

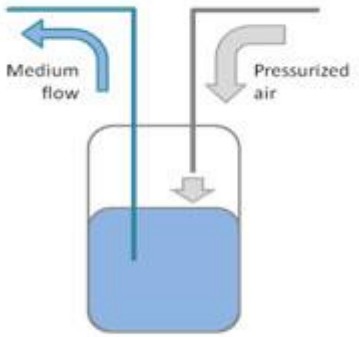


Figure 1 - Pressure induced perfusion scheme [7].

Peristaltic Pumps

A peristaltic pump is another technique to deliver a controlled flux of fluid to a culture chamber. The fluid flows into a flexible tube that suffers successive compressions and distensions to normal size in order to allow the fluid movement. A rotor connected to some rollers (2-8 rollers) is placed interiorly to tube and cause the mechanical deformation in the tube mentioned before. When the rotor moves the compressed section of the tube, the fluid is forced to enter the tube. Then, when the tube compression is released, the fluid exits the peristaltic pump.

This system can be easily sterilized since the unique component in contact with the fluid is the tube and the flux is controlled by the rotor angular speed. Furthermore, in this system the pump doesn't limit the volume quantity that can be send to culture chambers.

However, some compression triggered by rollers isn't so well coordinated and the tube sometimes occludes, causing an output pulsatile flux that is not desired. In order to achieve a pulse-free flow, some peristaltic pumps use spring-loaded rollers that avoid the occlusion of the tube.

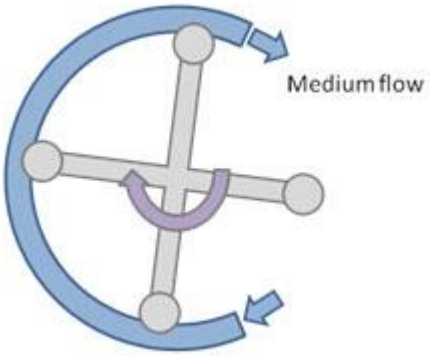


Figure 2 - Peristaltic pump scheme [7].

Syringe Pumps

In syringe pumps, the fluid is inserted inside a syringe. A motor with a specific angular velocity will control the force applied in the syringe plunger during a given period of time, which will determine the flow rate and dispensed volume desired.

One advantage of syringe pumps is the fact that the user can adapt the range of the instrument by changing the dimensions of the syringe.

The displacement of the piston and the volume injected are correlated, since a minimal movement induces a minimal injected volume. The flow stability is determined by the minimal movement of its motor and oscillations or pulses might appear at low flow rates due to the motor step. In turn, the volume injected is also dependent on the syringe diameter. Thus, a small syringe leads to low volume injection and, consequently, to a better flow stability at low flow rates. However, it limits the flow range, and thus, it must be weighted the commitment between flow stability and its range, depending on the syringe diameter. Further, the volume to be dispensed is limited by the maximum volume capability of the syringe.

Figure 3 - Syringe pump scheme [7].

2.2. Microscopy fluid delivery technology survey

2.2.1. Commercial solutions

In this section, it will be presented some fluid delivery system available on the market for microscopy applications. Commonly, these are quite expensive products so this review will be focused on specifications and practical capabilities. This analysis will be useful to later on define features for the device to be created, in order to cover at least slightly the ones that these products present.

2.2.1.1. Pressure induced perfusion

ELVEFLOW is a French company that researches and sells products for "high performance and Plug and Play flow control systems fitted to microfluidic research".[10]

One of the system configurations they purpose is represented in Figure 4. As it can be observed, the fluids are delivered by the pressure-driven perfusion method. A pressure generator supply the pressure controller (in this case ELVEFLOW OB1 Mk3) that has some independent channels that can be controlled by a personal computer (proper software sell by ELVEFLOW) in order to adjust the output channels. When the pressurized gas enters the reservoir, the fluid is expelled from it and will perfuse, in this case a microfluidic chip. [11]

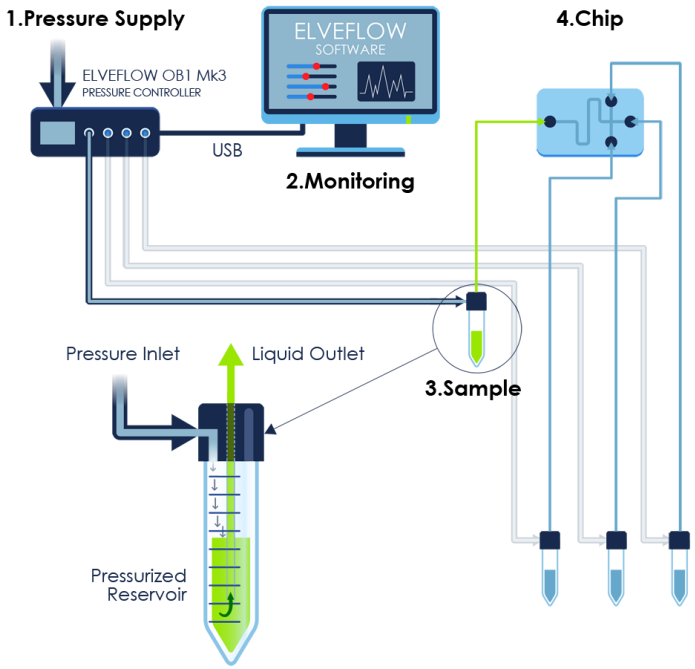


Figure 4 - Fluid delivery system proposed by ELVEFLOW [11].

They propose also the control of the output flow of liquid from reservoir through a flow sensor after it (Figure 5). The data collected should be sent back to pressure control that will adjust the output pressure in order to correct the liquid flow desired by the user, to be perfused using a proportional-integral-derivative control.[11]

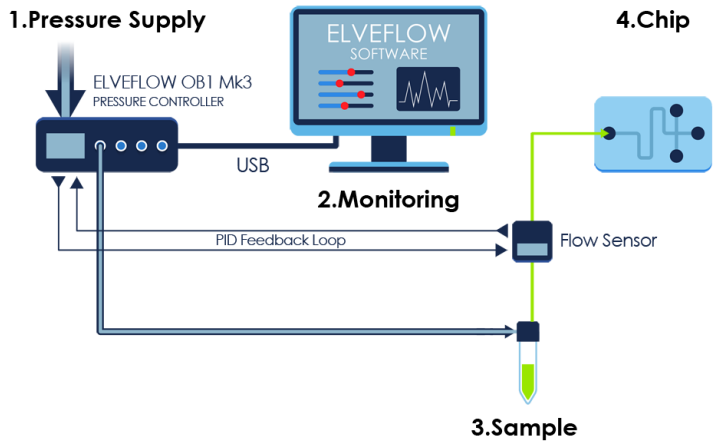


Figure 5 - Fluid delivery system with feedback control proposed by ELVEFLOW [11].

Other products available to purchase are their microfluidic flow switch matrices. They are ON/OFF valve matrices controlled through software that allows fast flow switching. That may be interesting since they allow drug and cell medium quick switching. [12]

Figure 6 - Valve matrix by ELVEFLOW [12].

Other option might be Ibidi, which is a "supplier for functional cell-based assays and advanced products for cellular microscopy". They sell a great variety of different products, from slides, dishes, microfluidic plates, cells and reagents to instruments, accessories and software for several applications.[13]

They developed a pump system for cell cultivation under flow, constituted by an air pressure pump, a fluidic unit and pump control software. The system schematic is represented in Figure 7.

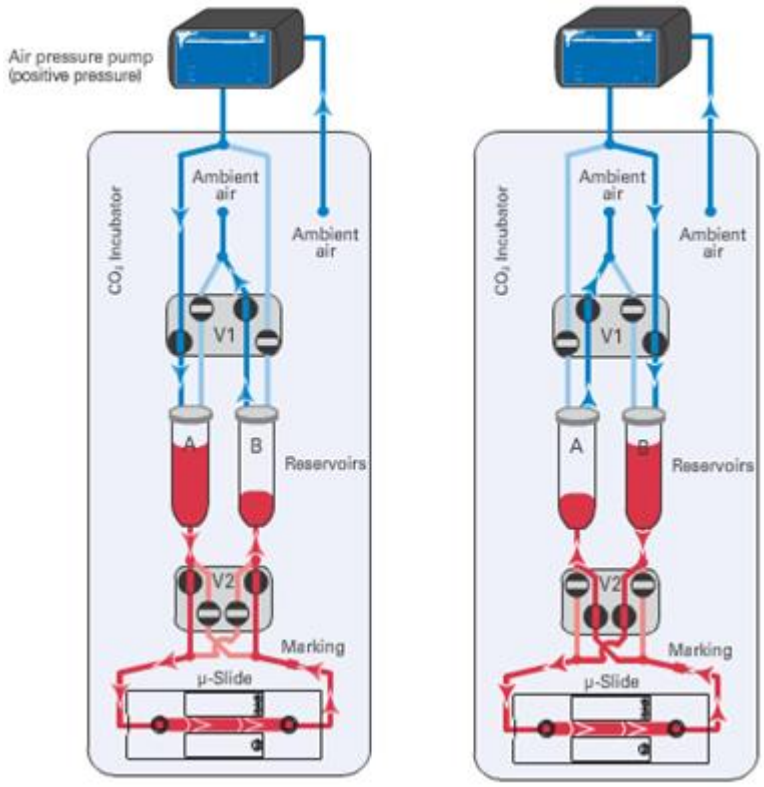


Figure 7 - Ibidi system - mechanism of working [14].

The fluidic unit has two reservoirs and a software controlled valve system that allow fluid movement. Initially, the valves state let the fluid travel from reservoir A to B through the µ-slide (Figure 7(a)). Then, the valves state changes and the fluid travels in the opposite direction, from reservoir B to A, but maintaining the flow direction through the µ-slide (Figure 7(b)), allowing a continuous unidirectional fluid movement. The main advantage of this system is the recirculation of the fluid between the two reservoirs since it reduces the minimal amount of medium and supplement needed. However, if the purpose of the experiment suggests a directional oscillating flow, the valve state can be reconfigured to perform such kind of behaviour.[14]

Other example is the CellASIC® ONIX Microfluidic platform. It is an easy-to-use and intuitive platform solution, whose control system is connected to the microfluidic plate via a low-profile manifold, which enables setup on any inverted microscope (Figure 8).

The microfluidic plate has four independent culture chambers (in parallel) and in each chamber can be delivered up to 8 different reagents such as culture medium or drugs. The plate is totally vacuum sealed with an upper plate piece (manifold). At the end of the experiment, the chamber's content is removed to a waste well (1 for each chamber), in order to be able to perform a new experiment without opening the plate. The upper plate piece is wired connected to the control system device.

The control system performs internal pressure/vacuum supply to enable flow control in any setting, regulates it, switch between open/close state of valves and controls CO2 and temperature inside the plate.

It includes a control software that allows intuitive controlling of pressure driven flows and also flexible user-defined flow programs, through a PC.[15]

Figure 8 - CellASIC® system overview [15]

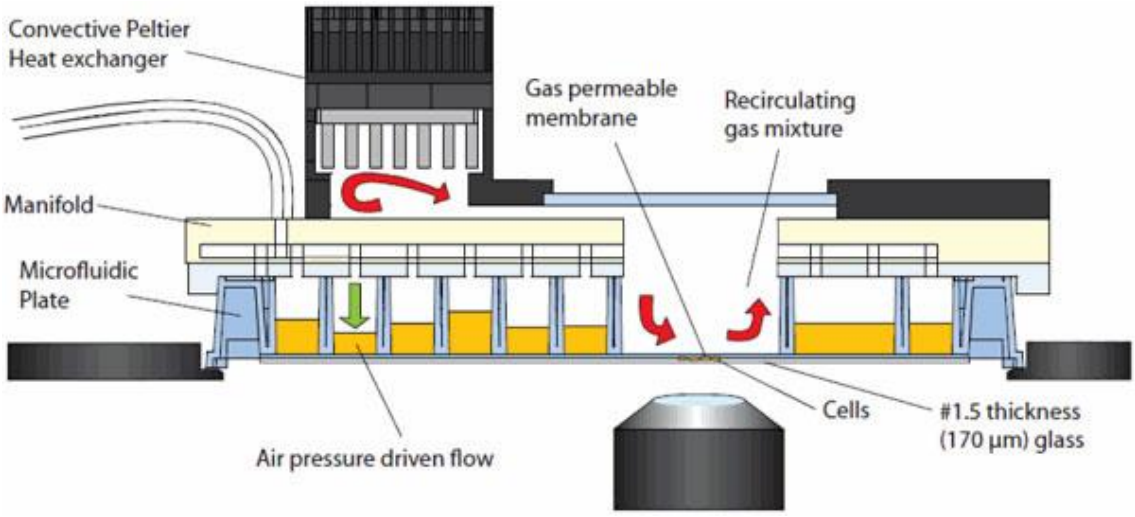


Figure 9 - CellASIC® system - working mechanism [15].

2.2.1.2. Peristaltic Pumps

There are several peristaltic pumps in the market with different characteristics. Therefore, this section pretends to review some examples of these product and their main characteristics.

Table 1 - Peristaltic pumps comparison.

	MINISTAR Peristaltic Pump [16]	Peri-Star Pro 2H / 4H [16]	Peri-Star Pro 4L / 8L [16]
Number of rollers	4	4	8
Number of channels	1	2-4	4-8
Rotor speed range (rpm)	1-50.0	1–100	1–100
Flow rate range (mL/min)	0.06-14.0	0.8–280	0.01–80
Tubing range (mm)	0.8~1.0 Wall Thickness ≤ 4.8 Outer Diameter	3.1–6.4 inner diameter	0.5–2.4 inner diameter

These three examples could have been selected for this project perfusion purposes (culture medium change and drug delivery) since all of them can deliver micro amounts of fluid. However they differ essentially on four properties: the number of rollers, the number of channels, the flow rate range and the tubing range. In this kind of experiences, normally low rotor speeds are used in order to have a better control of the amount of volume that is being delivered.

The number of rollers is a good parameter to take into account when the purpose is to reduce the characteristic pulsation flow of peristaltic pumps. Increasing the number of rollers will let reducing the pulsation amplitude but increase its occurrence frequency.

Depending on the executed protocol, multichannel can be also a better option when performing replicative and comparative assays, reducing the number of peristaltic pumps needed.

At last, flow rate range is closely related with tubing dimensions. These pumps accept a range of tubing dimensions that will compromise the flow rate range. Small diameter tubes will allow to deliver flow at a smaller rate, and consequently to deliver smaller amount of fluid volume.

In this case, Peri-Star Pro 4L / 8L would be preferable for small volumes deliver as drugs perfusion because it can host smaller calibre tube allowing lower flow rate (Table 1). The two other pumps (MINISTAR Peristaltic Pump and Peri-Star Pro 2H / 4H) can be suitable, for instance, to medium change since they operate with higher calibre tubes. [16]

2.2.1.3. Syringe Pumps

There are a great number of companies developing and selling syringe pump for microscopy applications. Two different models were selected to analyse their specifications and capabilities, one from Harvard Apparatus and other from World Precision Instruments (WPI).

Table 2 - Syringe pumps comparison.

Pump Model	Harvard Apparatus PHD ULTRA™ [17]	WPI SPLG270 [16]
Number of Syringes	2 to 10*	2-6*

Minimum Syringe Size (µL)	0,5	0,5
Maximum Syringe Size (mL)	140	140
Minimum Flow Rate (pL/min)	1,56	5
Maximum Flow Rate (mL/min)	220,97	215,803
Average Linear Force (lbs)	75 (Adjustable)	75
Programmable	Yes	Yes

** Depends upon the Syringe rack*

Both models share the same size for the syringe they can host in the device, and depending on its size, the flow rate range will vary. For smaller calibre syringes, it'll be able to produce lower flow rates, and the opposite occurs for higher calibre or lower length ones. The flow rate is also limited by the step motor resolution. In literature, it was found that for WPI syringe pump it is used a motor with 6400 microsteps per revolution.

Both pumps can vary their functioning characteristics depending on the upgradable accessories associated. They can infuse, withdraw, infuse and withdraw separately, or infuse and withdraw simultaneously depending on the experience's purpose. Furthermore, the syringe rack linked to the device can vary depending on the pretended number of syringes. However, since only one motor press multiple syringes, all of them will deliver the same amount of volume. Thus, these systems are suitable, for example, for multiwell plates when during the lab experience, a protocol step involves the same volume delivery for each well (e.g., medium replacement). However, they are not so suitable for different volume amounts delivery (e.g., chemotaxis assays with different amounts of drugs). This problem can be overcome with an auxiliary valve system, whose control will define in which well the fluid is being delivered.

These companies have also their own software in order to make these devices more flexible, allowing the users to configure and run a limited number of programs with a limited number of steps. [16][17]

2.2.1.4. Other pumps

Besides the pump types mentioned before, which are the most commonly used for microscopy applications, other kind of pumps were searched in order to be aware of possible solutions that would work for this purpose. The main difficulty was finding other kind of pumps that would satisfy the small flow rate ranges that this pumps have to admit.

One type of the pumps that would be suitable was the micro annular gear pumps by HNP Mikrosysteme. In these kind of gear pumps, two rotors are eccentrically placed and they have different properties: one external spur gear with less number of cogs and one internal spur gear (cogs faced inward).

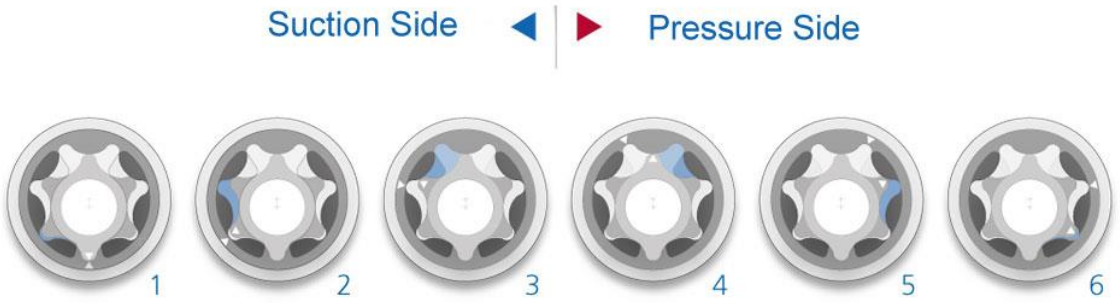


Figure 10 - Micro annular gear pump - gearing cycle [18].

The rotor placed internally can be driven by an external electrical motor as a stepper whose characteristics (mainly steps per revolution) will define the flow rate range. Consequently, the rotor with inward-faced cogs is driven by the internal one. Their gearing cycle is synchronized as you can see in Figure 10 where after a cycle, the cogs will fit again as they fitted in position 1.This pumps deliver a constant amount of fluid for each revolution called displacement volume, have a nearly pulseless flow and may be able to operate in the opposite direction allowing infuse and withdraw. [18].

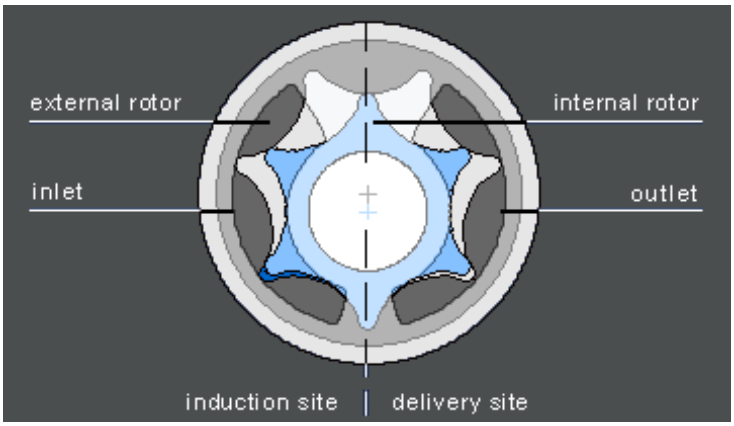


Figure 11 - Micro annular gear pump components [18].

As it can be seen in the table below, these companies have already high quality products in the market to deliver small amounts of volume ranging until tenths of µL [19].

Product	Vg (µL)	Minimum flow rate (ml/min)	Maximum flow rate (ml/min)	Minimum dosage amount (µL)
mzr-2505	1,5	0,0015	9	0,25
mzr-2905	3	0,003	18	0,5
mzr-4005	6	0,006	36	1
mzr-4605	12	0,012	72	2

Table 3 - Micro annular gear pump examples by HNP Mikrosysteme.

Dolomite is the world leading company that sells microfluidics products and accessories. In addition to pressure driven, peristaltic and syringe pumps, they also advise performing microfluidic applications with electro-osmotic and piezoelectric ones.

In electro-osmotic pumps, an external electric field applied on an electric double layer (EDL) generates the pumping mechanism able to produce high flow rates at high pressures. They are a good option to constant pulseless flows. Furthermore, they are very compact and lightweight, have low power consumption [20].

Figure 12 - Electro-Osmotic pump example [20].

Lastly, due to their compact dimensions and low energy requirements of piezo elements, piezoelectric pumps are also being applied to continuous fluid delivery. Basically, these pumps have a diaphragm with a piezoelectric actuation. The piezoelectric component (normally a disk) is placed over the diaphragm and when actuated, it deflects the diaphragm. Furthermore, these systems have also an associated valve system, which let the main system reach a unidirectional flow avoiding reflux that would reduce pump efficiency. Since the volume being pumped per cycle is small, a wide range of flow ranges can be achieved adjusting the voltage and frequency applied to piezoelectric actuator. Depending on pump chamber size, small amounts of fluid can be delivered ranging from milliliter to picoliter. In table A are some specifications of three piezoelectric pumps by Dolomite.[21]–[23]

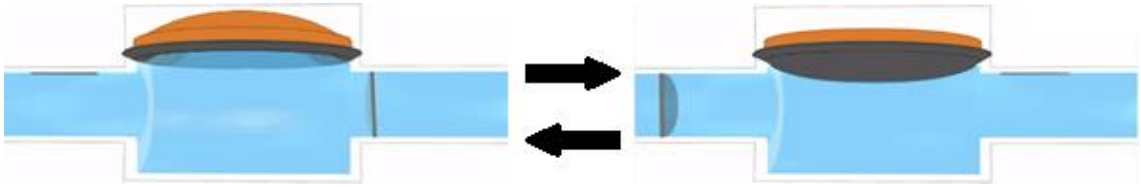


Figure 13 - Piezoelectric pumps mechanism [23].

Table 4 Dolomite piezoelectric pump specifications (OD – outer diameter; ID – inner diameter; L – Length) [28].

Specification	Piezoelectric Pump 3ml/min	Piezoelectric Pump 7ml/min	Piezoelectric Pump 20ml/min
Maximum flow rate (mL/min)	**3**	**7**	**20**
Maximum pump pressure (kPa)	**40**	**45**	35
Applied frequency range (Hz)	**10 - 60**		
Tubing barb size (mm)	OD 1,2; ID 0,6; L 2,5	OD 2,2; ID 1,2; L 3,5	OD 2,8; ID 1,8; L 5,0
Compatible tubing ID (mm)	0,79 – 1,0	1,59 – 2,0	2,0 – 2,38

2.2.2. Related work

The problem related to expensive and closed lab products affects the possibility of worldwide research groups to perform new kind of experiments that would bring advantageous results. However, there are already some researchers who tried to overcome this challenging problem and design, produce and test “do it yourself” (DIY) equipment. The following sections will focus on four different approaches that were performed in order to develop devices with similar aims of this project.

StreyLab, a bioengineering research group from Stony Brook University, manufactured their own fluid delivery system for microfluidic applications. They build an Arduino shield that is able to control 8 fluid channels, each one with an air pressure controller and an air valve. In order to produce an analogue output to air pressure controller, they attached digital-to analogue converters (DAC) and a darlington array [24] to drive the air valves. The final aspect of the assembled Arduino shield and the pressure controllers array can be seen in Figure 14 and Figure 15, respectively.

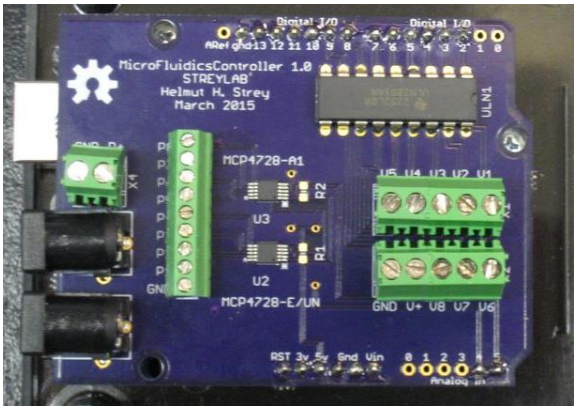


Figure 14 - Arduino Shield (ULN1: Darlington array; V1-V8: air valves output; MCP4728: 4-channel DAC; P1-P8: air pressure controller output; Barrel jack) [24].

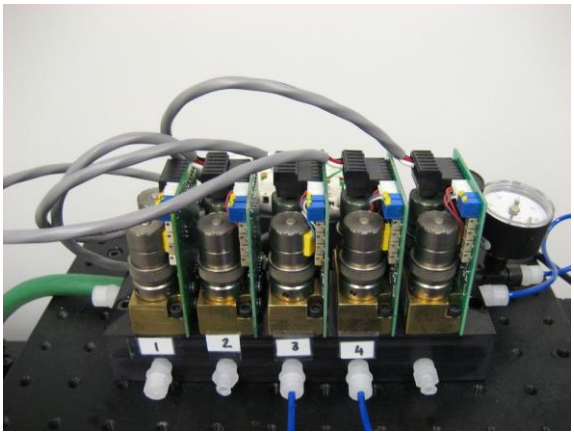


Figure 15 - Pressure controllers array [24].

Another system was created and tested by the Augsburg College‘s Physics Department, that built an automatic lipid dispensing device based on the Ardulink library, which is an open-source java solution for Arduino boards[25]. They mention that it could be used a syringe pump and software to control it, available on the market already, but the associated costs are high. So, they made some low-cost replacements. They used a µL solenoid valve, an adjustable stepper motor pump and a feedback sensor that measure the liquid already dispensed. A

stepper motor driver actuated by an Arduino board drives the pump. Furthermore, they used also an optical sensor to sense stepper motor position in order to give feedback to the system and readjust it allowing more precise dispensing.

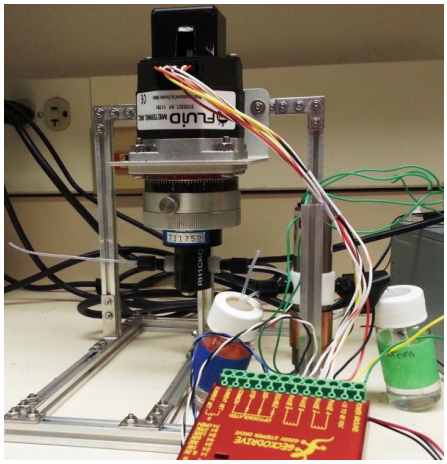


Figure 16 - Automatic liquid dispensing system [26].

It was also developed a Java application to interface with users (Figure 17). After establishing a correct connection with the Arduino board, the user can define the volume to be dispensed.

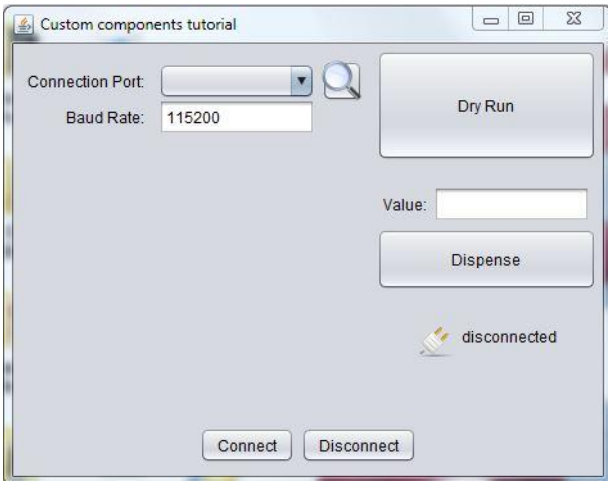


Figure 17 - Java application [26].

It was compared their system with a microliter syringe and found that their system is more repeatable since it admits lower variances. Further, they observed that these variances drastically increase with the amount of volume dispensed. They even compare the accuracy of both system questioning the reliability of expensive market solutions when compared with their accurate and cheaper self-made solutions (up to 60% price save estimation) [26].

Balakrishnan et al published a paper where it is described how they build a scalable perfusion culture system. They used 3D-printing to produce their own perfusion bioreactors and mini peristaltic pumps. For bioreactors, they build a base, a body and a cap with VeroWhite 3D-print material (Figure 18). The assembling scheme can be seen in Figure 19. A Polydimethylsiloxane (PDMS) ring was used to cushion the coverslip over it, sealing the bioreactor in the base component. The cap was assembled afterwards without a coverslip since they wanted also to be able to pipette into the bioreactor [27].

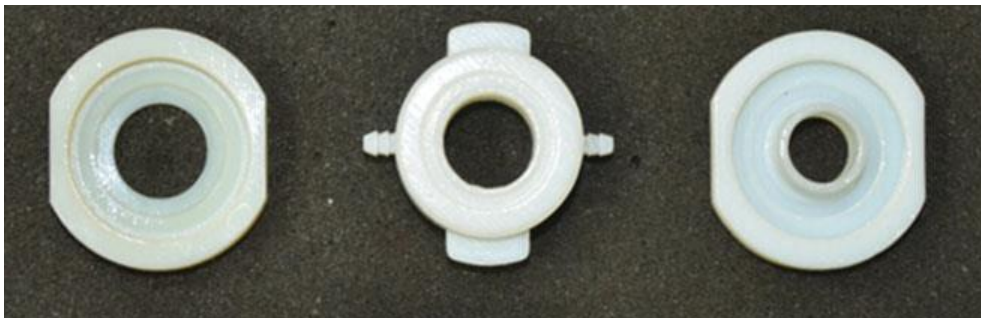


Figure 18 – Bioreactor components from left to right: base, body and cap [27].

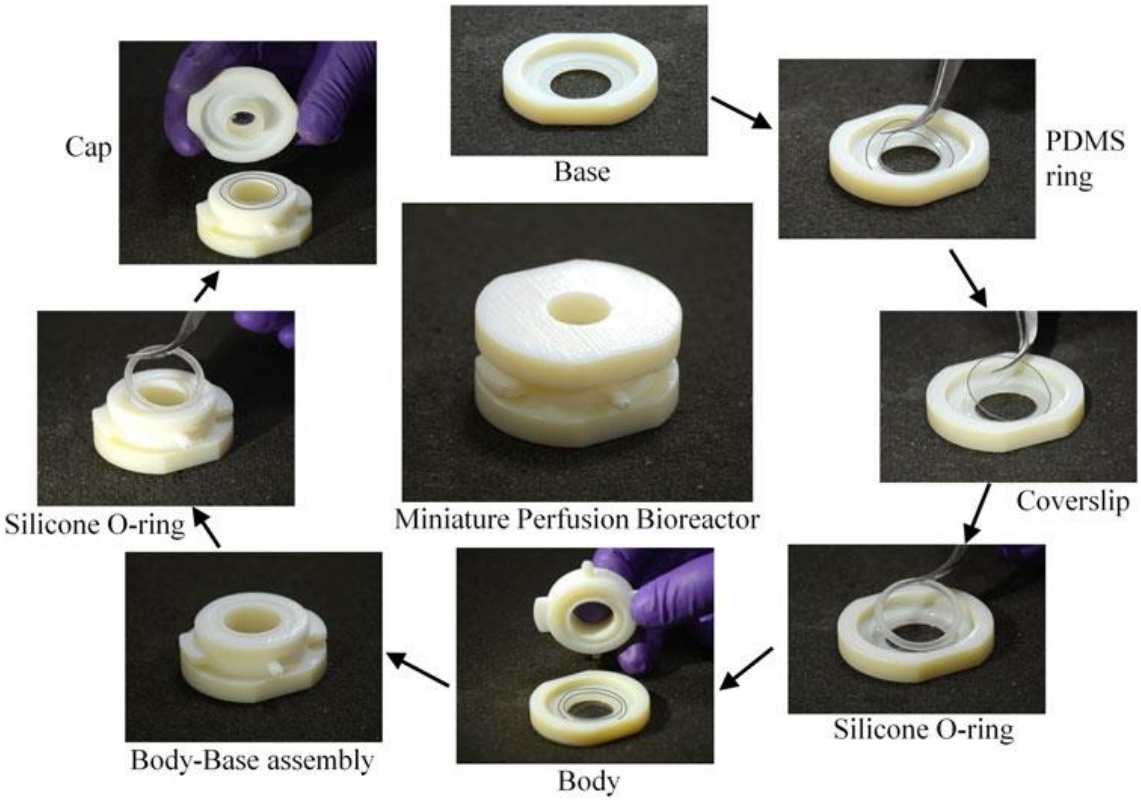


Figure 19 - Bioreactor assembling scheme [27].

The peristaltic pump manufacturing was also split in three components: the housing, the rotor and the casing (Figure 20). A ball bearing was firstly placed inside the housing in order to fit with the rotor connection. The tube was inserted through two holes in the housing top layer. A DC motor was put from behind connected to roller allowing it to move.

Figure 20 - Peristaltic pump components: housing, rotor and casing from left to right [27].

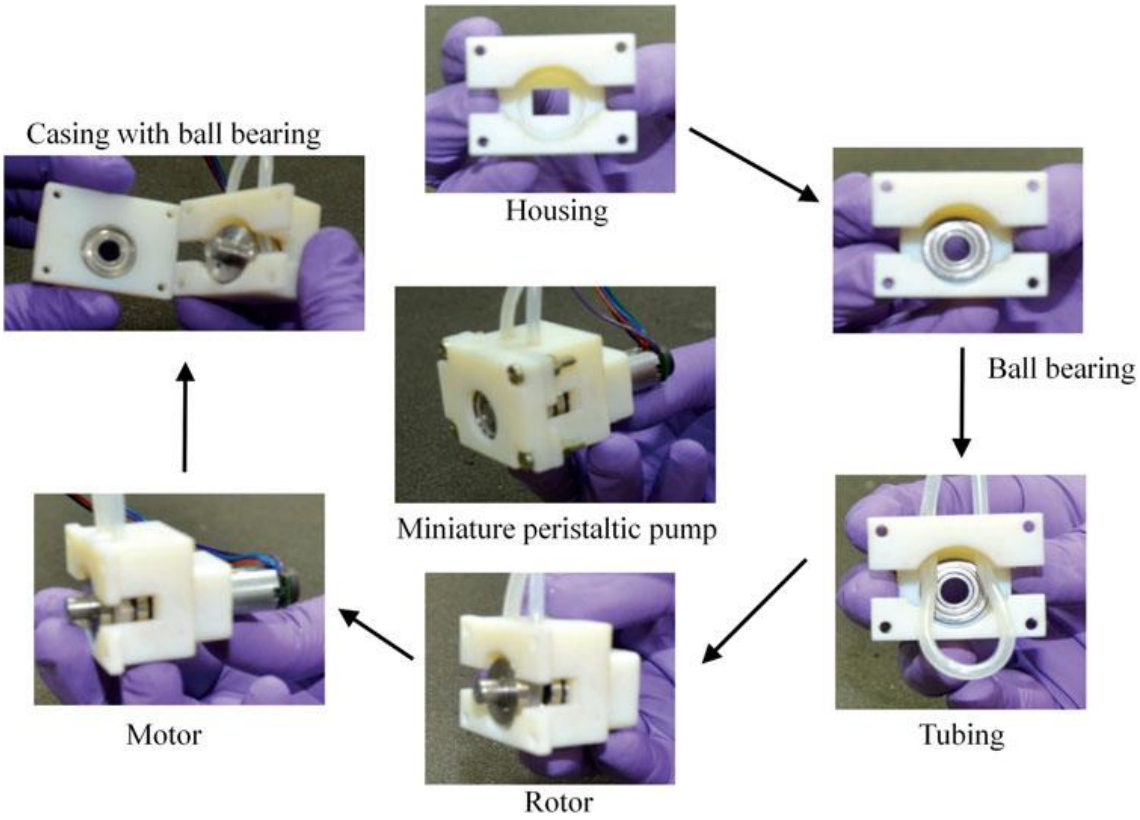


Figure 21 - Peristaltic pump assembling scheme [27].

At last, they assembled a closed loop perfusion system with a reservoir, the bioreactor and the peristaltic pump and validated their flexible system since it can be used for several application as live cell imaging and perfusion cell culturing.[27]

Michigan Tech developed another project in Open Sustainability Technology (MOST) research group. They see the open-source material (hardware and software) as a valuable opportunity to easily reduce costs and the labware is one of their major focus since they are aware of how extremely expensive they are for research institutions.

Therefore, they managed to produce an Open-Source Syringe Pump controlled by a Python program, running on an open source Raspberry Pi computer. The syringe plunger movement is controlled with a stepper motor driven by a stepper motor driver.

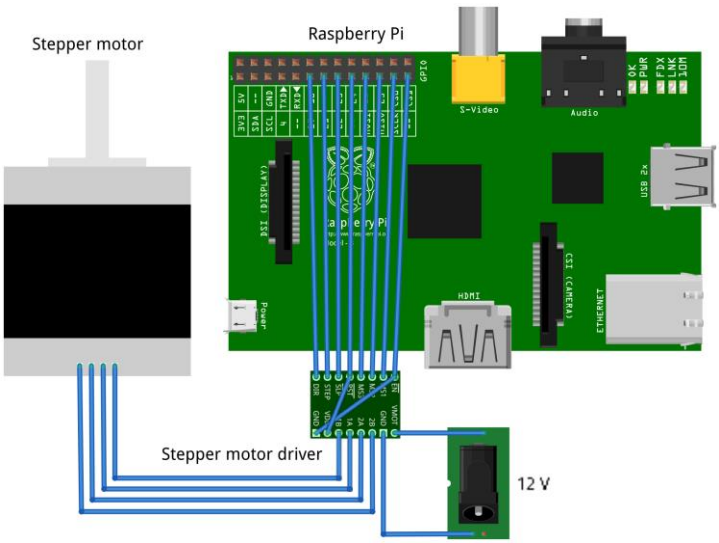
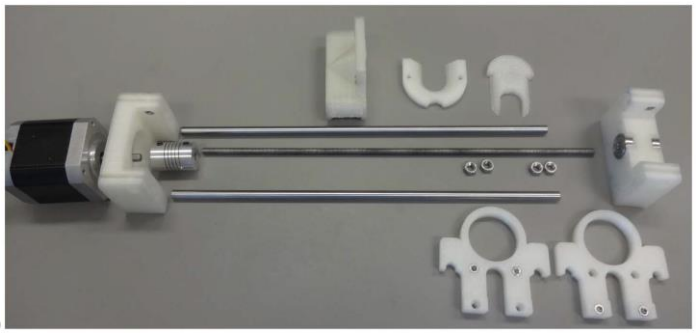


Figure 22 - Eletronic components of syringe pump system [28].

Almost all support components to hold the stepper motor driving the syringe movement were designed in OpenSCAD, open-source 3D-modelling software, and printed with RepRap 3D-printers, which are also open-source.

a)



b)

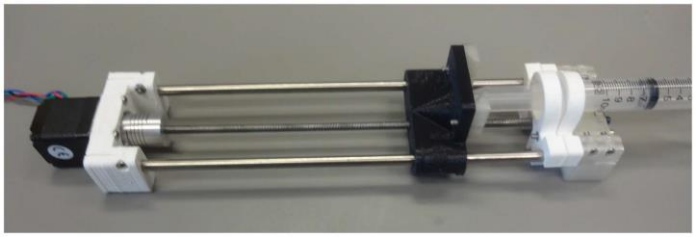


Figure 23 - Support components of syringe pump system (a) disassembled and (b) assembled version [28].

Three different versions were built, depending on the motor used and the number of syringes. For each model, they found the maximum flow rate and its accuracy, and posteriorly compared to other products available on the market. Furthermore, they advise that although the specifications that they have reached for their systems, this system is fully customizable by changing the motor characteristics and syringe size, depending on the desired final application. [28]

2.3. µManager - open-source microscopy software

The integration of different microscopy hardware (microscopes, cameras, shutters, etc.) is a really important step in microscopy experiments. This process is ensured by microscopy software. Microscopy companies which always try to sell their own microscopy control software when selling hardware. However, when using different devices from different companies, sometimes it is not possible to use both apparatus synchronized since, usually, they are not compatible. Therefore it is necessary to produce software that does not compromise performing multi-hardware microscopy and that enable, as much as possible, all kind of possible usable devices in microscopy applications. This is the purpose of Micro-manager.

Micro-Manager is an open-source software to control microscopy application. Originally, it was a plugin of ImageJ, which is a public domain Java image processing program that is widely used by scientific community. Essentially, it is an interface that allows the programming of microscopy-based experiments (scripts creation) and running it afterwards. Since it supports a high number of devices drivers, a lot of microscopy devices can be integrated, from almost all microscopes available on the market to cameras, stages, filter wheels, shutters or even serial port communication devices [1].

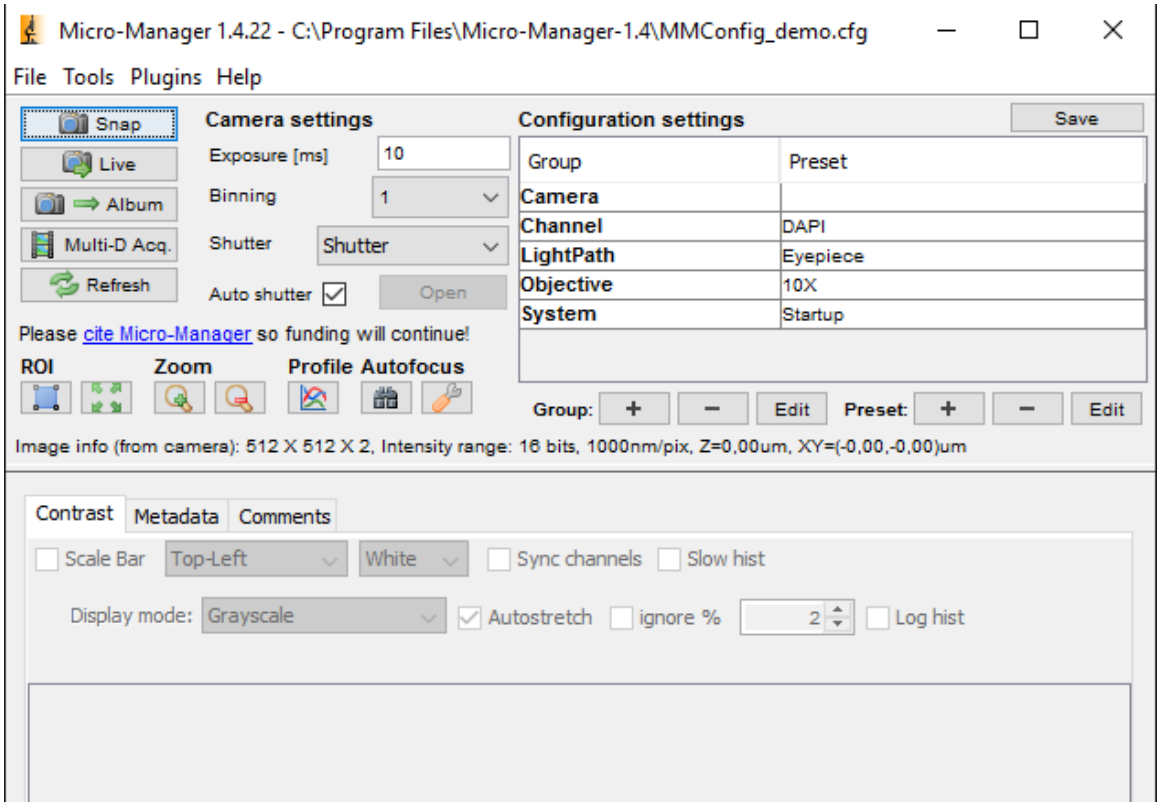


Figure 24 - Micro-Manager graphical interface.

The latest version is 1.4.22 and it is multi Operating System (OS) compatible. The source code is available to download at official µManager webpage[1].

The software architecture is split in three main layers: Graphical User Interface (GUI), Core Services (MMCore) and Device Adapters. The main layer is the "MMCore" module, which controls and synchronizes different devices. This is done including these devices plug-in modules. Then, the MMCore interface can be accessed from many different programming

[1] www.micro-manager.org

environments. This flexibility allows writing scripts based on an independent hardware language. Furthermore, this software is extensible since it is open-source and allows users to edit it and share new software capabilities [29].

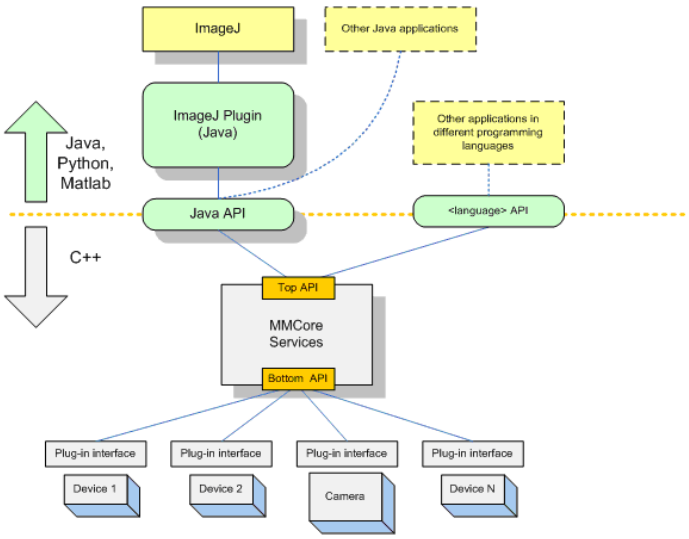


Figure 25 - Micro-Manager Software Architecture [29]

This software includes a Multi-Dimensional Acquisition Tool. This is a powerful engine that let the user performing a controlled image acquisition depending on several variables such as time, multiple stage positions, multiple slices or multiple channels.

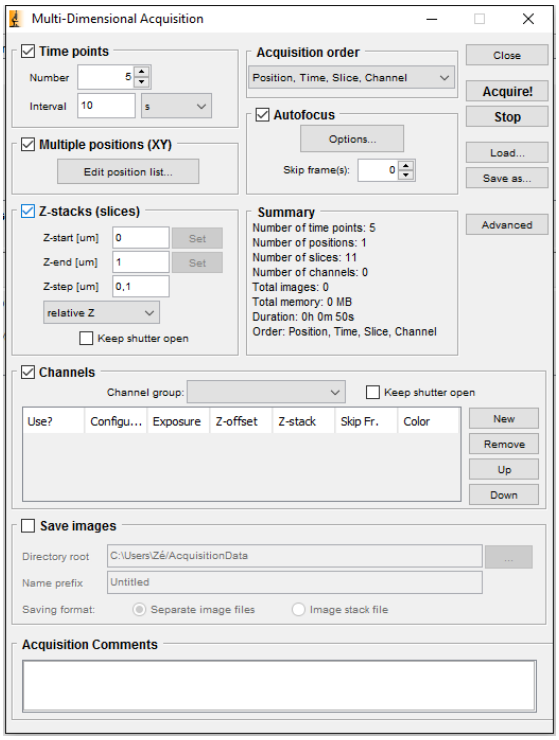


Figure 26 - Multi-Dimensional Acquisition tool window.

One of the devices already supported by micro-manager is Arduino. They provide a firmware to run on Arduino that will let it to communicate with Micro-Manager software through a serial port. That firmware was initially developed to control an acousto-optic tunable filter (AOTF) but it also provides sending and receiving digital commands with the software. Before start using the Micro-Manger, the hardware that will be used must always be configured

and the setup configuration can be saved. The support website recommends a hardware configuration and explains how the solution can be tested.

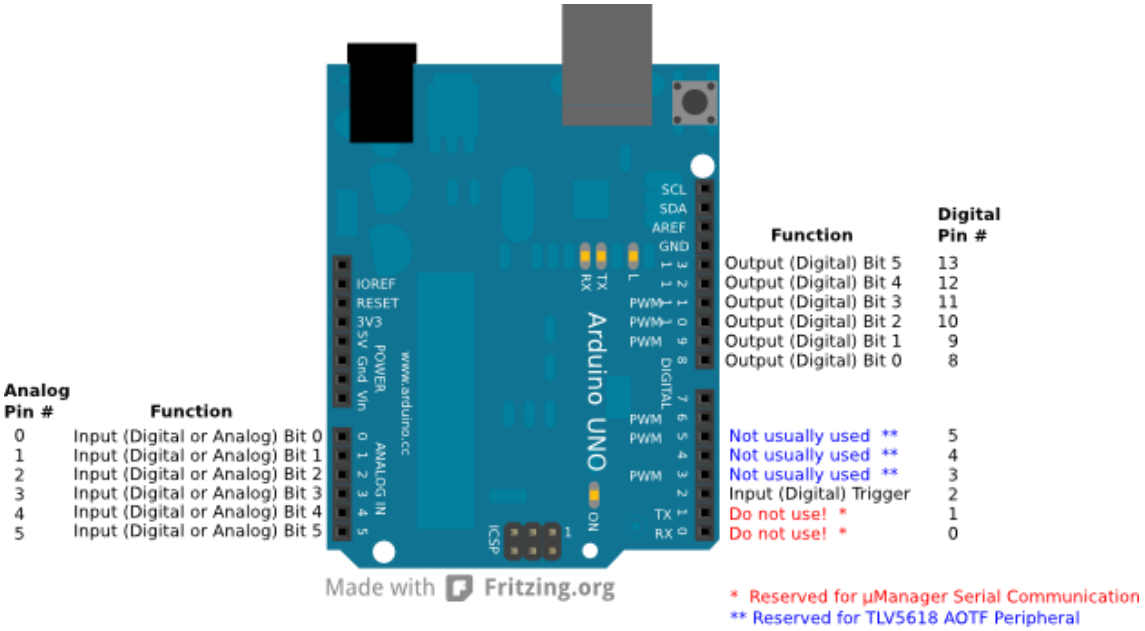


Figure 27 - Example of Arduino pin connections to Micro-Manager software.

Chapter 3

System design

3.1. System Architecture

This projects aims to create a viable and low-cost solution to infuse and withdraw accurate amounts of volume to a culture chamber in a controlled way. The procedures must be controlled through the Micro-manager software environment.

First of all, to perform a fluid delivery system, it was selected the pump type. Although the volume capability limitation, the syringe pump configuration was selected instead of peristaltic pump or pressure induced perfusion. Pressure induced perfusion pumps are able to perform perfusion tasks with higher stabilization levels but it requires to acquire a vacuum machine, which could not fulfil the low-cost requirement of the project. On the other hand, peristaltic pump option was also abandoned mainly due to its pulsing flow. Syringe pump seems to be the best option since the main disadvantages found (syringe finite volume) would be easily overcome because the volume amounts that will be exchanged can fit several times in syringe maximum volume capability.

The user may be able to select from three main pump task types: medium change, simple fluid delivery and calibration procedures as it can be seen at the use-case diagram below (Figure 28).

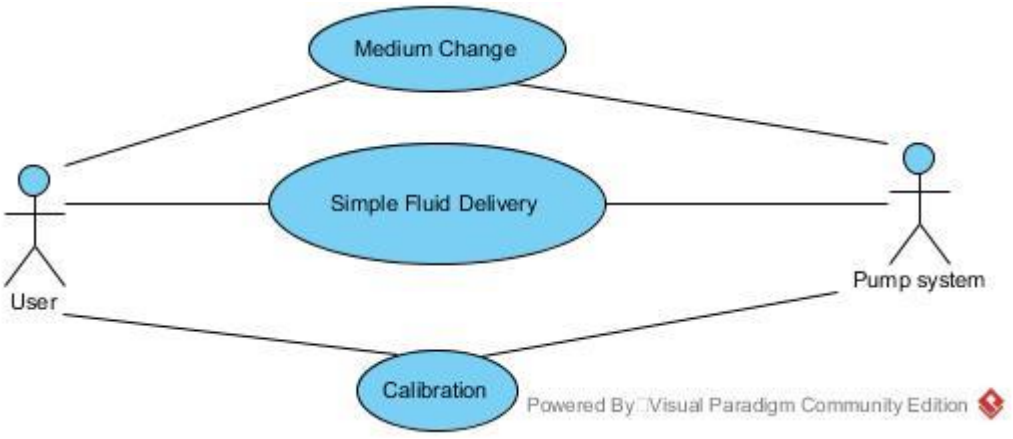


Figure 28 - Use-case diagram

It was defined that communication between Micro-Manager software and pumps operation would be intermediated by a microcontroller. An Arduino board was used since there

are already device drivers that enable the communication between this microscopy software and the Arduino by COM serial port. Thus, this communication protocol was analysed and new capabilities were added, namely to manage syringe pump tasks.

As mentioned in section 2.2.2., there are two kinds of cell culture: static and perfusion cultures. When working with perfusion cultures, a fluid delivery system is necessary to automatically perform the demanded fluid exchanges. Thus, at this initial stage, it was decided to test the fluid delivery prototype with static cultures since currently they still the most commonly used cell culture technique and they are much easier to handle when compared with perfusion cultures. The culture chamber selected included a 35 mm diameter µ-Dish since it is a cheap and single culture chamber.

Taking into account a range of possible applications, several system requirements were identified.

The first requirement consisted in overcoming the syringe finite capabilities so they cannot represent a limitation for the final solution. Hence, some modifications had to be performed to the 3D-printed components.

It was checked what volumes the syringe pump would be able to infuse/withdraw. The µ-dishes selected have a full volume around 11 mL for high wall ones and around 12 mm height without lid. However, it is recommended not to fill the 35mm µ-Dish more than 2 mL because high amounts of volume can lead to liquid contacting with the lid. Furthermore, the drug volumes to be infused should not be greater than 100 µL hence, the syringe pumps should fit syringes ranging from 1 mL to 50 mL of total volume. The smaller ones are suitable for drug delivery, and the bigger ones for medium change, allowing several medium replacement procedures during a single experimental protocol.[30]

The user should control a maximum of three syringe pumps connected to a 35 mm diameter µ-Dish culture chamber from the Micro-manager software, not only from the script environment but also from a user-friendly interface. This GUI should allow easily inserting a pump task sequence to run during image acquisition. Hence, a new tool should be added to the multi-dimensional acquisition engine, since the capability to perform a pump task sequence while acquiring an image sequence is seen as a valuable appliance by the final user. Thus, Micro-Manager software should be enhanced, not only the Arduino device driver C++ module but also the Java layer. The device driver enhancement is sufficient to control the syringe pumps through the Micro-Manager script interface. However, some GUI components should be created and edited to achieve a more familiar and useful interface. Below, a scheme of the system is described in detail.

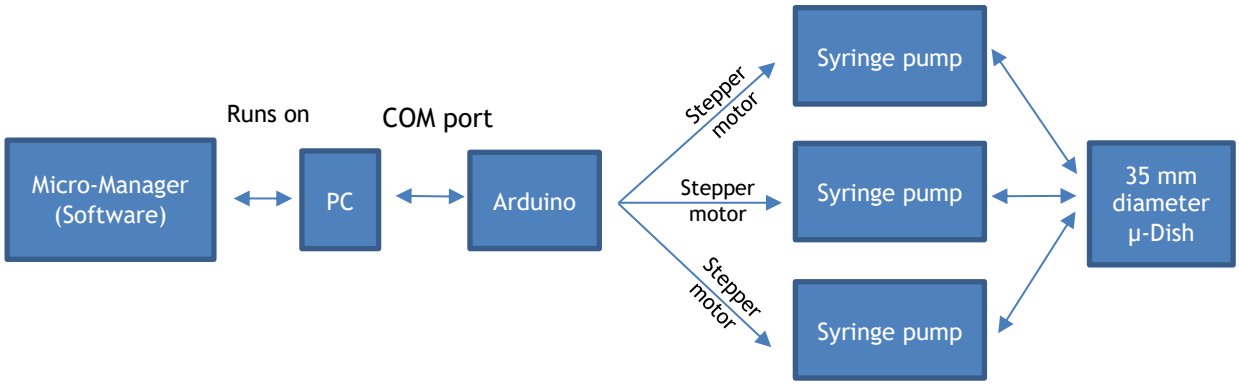


Figure 29 - System overview

3.1.1. Hardware

3.1.1.1. Syringe pump

The open source syringe pump designed by MOST research group [28] was selected to perform infuse and/or withdraw of fluids to cell culture chamber.

Since all its 3D-printed components source code is online available, it was easier to adapt and enhance this original solution to fulfil project requirements than creating a new syringe pump from scratch. Furthermore, they presented acceptable performance assessment results.

MOST syringe pumps are essentially composed by a stepper motor, components and fitting components such as screw, nuts, rods and shaft couplers. The shaft rotation produced by the stepper motor is converted into linear movement of the carriage that apply force on the syringe plunger, allowing the liquid infusion. The stepper motor can take a clockwise and anti-clockwise rotation. Hence, the plunger can run forward and also backward, allowing to perform infuse and withdraw tasks. The conversion of rotational movement into linear movement is trickily performed by a trap nut positioned into the carriage. The clamps hold the syringe body in order to not move relatively to the carriage. The friction force between contacting components are reduced using bearings, not only between the screw and the idler end but also between the rods and the carriage. All the components are connected using M3 screws and nuts namely the motor end 3D printed component attachment to the NEMA stepper motor.

Table 5 - List of material for one MOST syringe pump.

Materials	
3-D Printed	**Count**
Motor End	1
Carriage	1
Plunger Holder base	1
Plunger Holder tab	1
Body Holder	2
Idler End	1
Motors & Metal	**Count**
NEMA17 motor	1
5mm x 5mm shaft coupling	1
625z ball bearing	2
LM6UU linear bearing	2
M3 x 10mm socket head cap screw	6
M3 x 20mm socket head cap screws	4
M3 x 40mm socket head cap screws	4
M3 hex nut	13
M5 hex nut	5
M5 threaded rod 0,2 m	1
6mm A2 tool steel 0,2 m	2

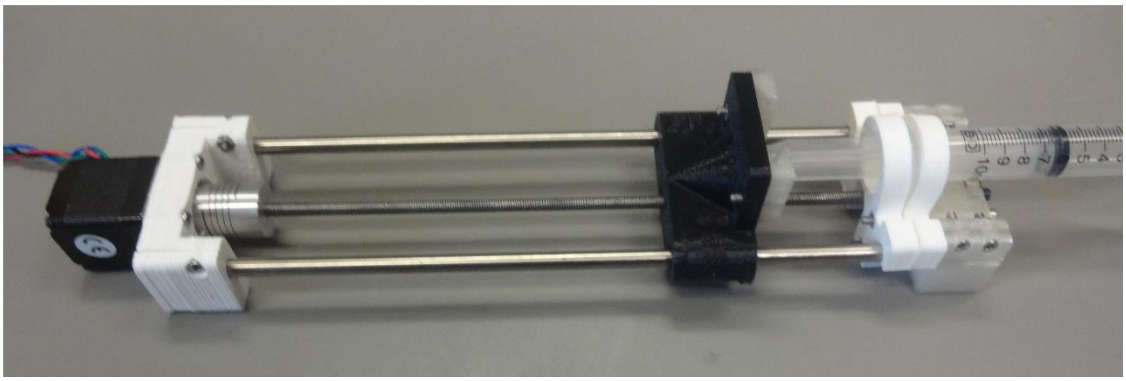
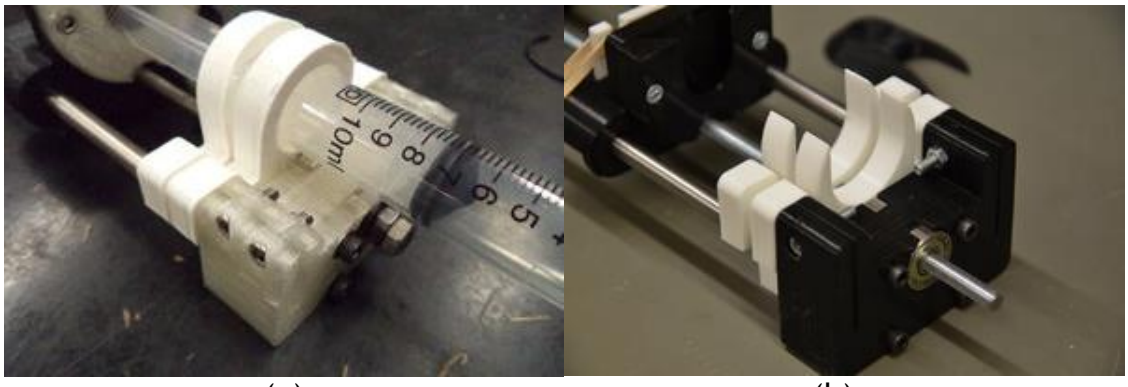


Figure 30 - MOST original Open-source syringe pump [28]

The MOST research group has also available online the list of materials, the assembling protocol and the OPENSCAD source code of the 3D printed components. This code was modified to fulfil the pre-established prototype requirements [28].

The original design of the clamp that hold the syringe body had already been modified by Lynch, being opened in order to allow easy removal of the syringe instead of the original closed one [31].



(a) (b)

Figure 31 - Clamp modifications performed by Lynch (a) Original MOST clamp (b) Lynch clamp modification [6][39].

This modification was also adopted since during an experimental protocol, it can be necessary to replace a syringe.

The first modification was performed on clamps. The concave clamps zone may vary its diameter depending on which syringe it is holding. This way, to hold the 5 syringe sizes that were purposed (1mL, 5 mL, 10 mL, 20 mL and 50 mL), it would be necessary to print 5 different clamp types, and two copies for each one. So, the solution found consisted in defining two clamps types that would fit more than one syringe size. So, the first clamp type has the 10 mL syringe outer diameter, fitting the 1, 5 and 10 mL syringes, and the second clamp type has a 50 mL syringe outer diameter, fitting the 20 and 50 mL syringes. The table described below was taken into account to establish these decision boundaries.

Table 6 - Clamp types boundary decision.

Would a syringe with x diameter fit in a y diameter clamp?			Syringe body diameter (x)				
			1 mL	**5 mL**	**10 mL**	**20 mL**	**50 mL**
			0,65	1,41	1,71	2,14	3,05
Maximum syringe finger flange diameter (y)	**1 mL**	2,12	Fit	Fit	Fit	Do not fit	Do not fit
	5 mL	2,62	Fit	Fit	Fit	Fit	Do not fit
	10 mL	3,09	Fit	Fit	Fit	Fit	Do not fit
	20 mL	3,68	Fit	Fit	Fit	Fit	Fit
	50 mL	4,97	Fit	Fit	Fit	Fit	Fit

Furthermore, the original MOST syringe pump has a syringe plunger retainer that depends on syringe plunger diameter. This component fixes the syringe plunger, allowing it to follow the carriage movement forward and backward. However, since it depends on syringe plunger diameter, it would be also necessary 5 different types of syringe plunger retainers, one for each syringe type. Hence, a new plunger retainer had to be designed following the partition made before at clamp type's definition.

After performing the design modifications, all the components were printed in a BQ Witbox 3D-printer. Polylactic acid (PLA) was selected as filament and the printer were equipped with a 0.5 mm nozzle. The layer height resolution was 0,2 mm and the print speed was 60 mm/s.

3.1.1.2. Stepper motor and driver

In this syringe pump, the stepper motor capabilities are mainly related with the maximum volume resolution and the force produced on the syringe plunger.

MOST research group used a NEMA 17 stepper motor for their syringe pump.

Nowadays, the most common widely available NEMA 17 bipolar stepper motor has 200 or 400 steps per revolution. Better resolutions that the ones mentioned before are already available on market and they would lead to a better performance. However, it could lead to an abrupt increase of costs.

Furthermore, MOST research group estimated that the NEMA 17 motor they used developed 200 N. This value was taken as reference since the selected stepper motor should have similar mechanical properties. Further, small diameter plastic tubes should be attached to the syringe end (intermediated with a needle), in order to guide the fluid to the culture chamber. This diameter reduction relatively to the syringe body diameter needs a higher force applied on the plunger.[28]

Following the equation:

$$Torque = Force \times Lever\ arm \tag{3.1}$$

where lever arm refers to the perpendicular distance from force to the point of rotation and assuming a minimum 200 N of linear force and a shaft radius of 2,5 mm, the torque needed should be at least:

$$Torque = 200\ N \times 0{,}0025\ m = 0{,}5\ N.m \tag{3.2}$$

Hence, a NEMA 17 stepper motor with 400 steps per revolution and 0,48 N.m minimum holding torque was selected [32].

Figure 32 - 42BYGHM809 - Selected Stepper Motor [32].

To drive the stepper motor, a Pololu DRV8834 Stepper Motor Driver was used since it has a low voltage operating range and it is capable to fulfil the maximum current per phase allowed by the motor (1,7 A per phase). A maximum of 1,5 A per phase can be used without requiring additional cooling.

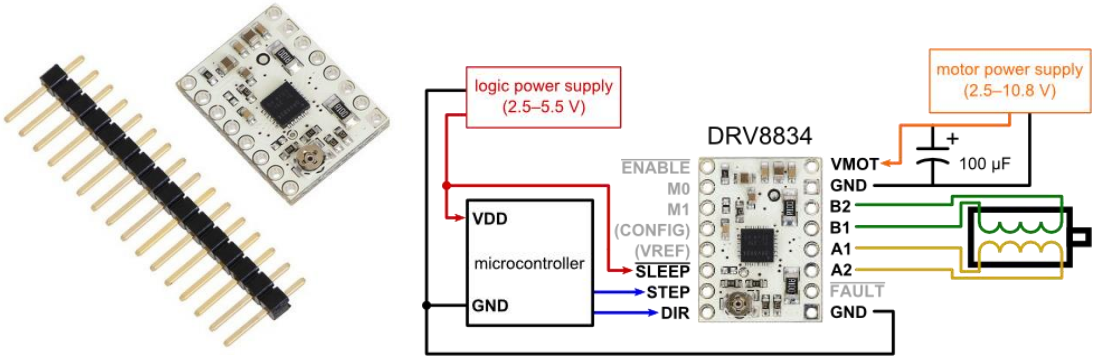


Figure 33 - Pololu DRV8834 Stepper Motor Driver and its minimal wiring diagram [33].

As it can be observed in Figure 33, this stepper driver receives two major inputs: STEP and DIR. STEP commands the stepper motor to perform one step and DIR defines the rotation direction. Furthermore, two other inputs can be received: M0 and M1, which set the microstep resolution. Some stepper drivers allow higher resolutions than the stepper motor resolution, allowing intermediate step locations that are commonly named as microsteps. This is possible by energizing the coils with intermediate current levels. This stepper driver has six different microstep resolution ranging from full step to 1/36 [33].

Table 7 - Pololu DRV8834 microstep resolution depending on M0 and M1 possible inputs [33].

M0	M1	Microstep Resolution
Low	Low	Full step
High	Low	Half step
Floating	Low	1/4 step
Low	High	1/8 step
High	High	1/16 step
Floating	High	1/32 step

This powerful feature enables the system achieving higher final stepper motor resolutions up to 12800 microsteps per revolution.

This stepper motor driver has also a potentiometer that sets the current limit. The stepper motor selected has a maximum power supply of 3,06 V, which depends on the maximum current per phase and resistance per phase, as stated by the Ohm law. Powering the motor with higher voltages than 3,06 V would lead to a current per phase higher than the recommended maximum, causing possible irreparable damages to motor. Hence, Pololu proposed the following equation to set the current limit:

$$Current\ Limit = VREF \times 2 \tag{3.3}$$

where VREF refers to the electric potential difference between the potentiometer and ground. The current limit selected was 1A per phase, so the VREF was adjusted to 500 mV. The current limit calibration should be performed before connecting the stepper motor to the driver. Furthermore, the driver should only be powered when all 4 coil wires of the stepper motor are connected or no one is connected. Intermediate wiring configurations while the driver is powered may damage the stepper motor [32][33].

3.1.1.3. Electrical circuit prototype

The final prototype circuit is essentially composed by an Arduino MEGA 2560, three stepper motors and the correspondent stepper drivers, an external power supply of 9 V, and a liquid crystal display.

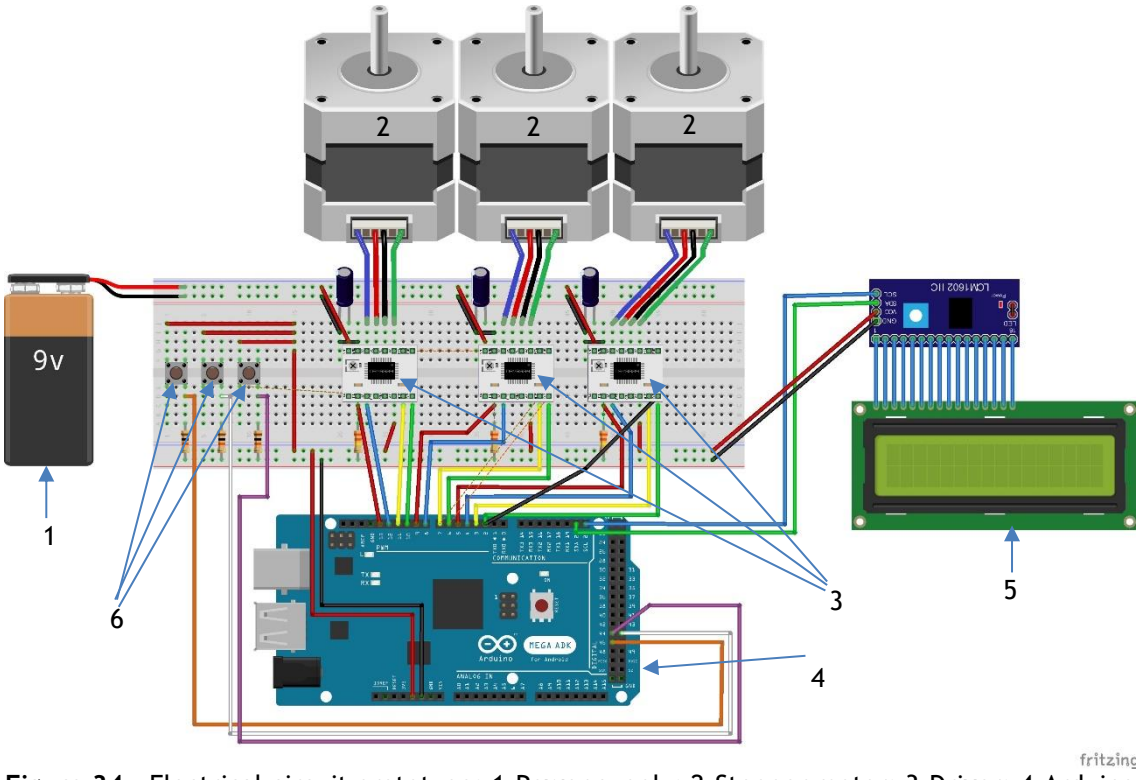


Figure 34 - Electrical circuit prototype: 1-Power supply; 2-Stepper motor; 3-Driver; 4-Arduino MEGA; 5-LCD; 6-Push buttons

The Arduino MEGA 2560 was chosen due to its higher number of pins, which allows managing a higher number of pumps. The inputs DIR, STEP, M0 and M1 of each stepper driver

were connected to PWM digital Arduino pins. Moreover, push-buttons were added in order to help the performance of some additional processes, like calibration tasks. A LCD with I2C connection was also added to perform debugging tasks and later on, to display valuable information such as the steps being performed to help performing manual tasks and debug the current system state. The pins are connected as described on the following tables.

Table 8 – Arduino pin connections

Arduino Pin	Connected to
2	DIR - stepper driver 1
3	STEP - stepper driver 1
4	M1 - stepper driver 1
5	M0 - stepper driver 1
6	DIR - stepper driver 2
7	STEP - stepper driver 2
8	M1 - stepper driver 2
9	M0 - stepper driver 2
10	DIR - stepper driver 3
11	STEP - stepper driver 3
12	M1 - stepper driver 3
13	M0 - stepper driver 3
20 (SDA)	SDA of LCD I2C adapter
21 (SCL)	SCL of LCD I2C adapter
44	Push – Button 1
45	Push – Button 2
46	Push – Button 3

Table 9 – Stepper motor wire connections

Stepper Driven output	Stepper motor wire
B2	Blue wire
B1	Red wire
A1	Black wire
A2	Green wire

3.1.1.4. 35 mm µ-Dish Lid

Assuming the application purpose of the developed system, it was defined µ-dishes as culture chamber. These dishes are usually completely sealed by a proper lid. However, this lid is not suitable to perform the intended fluid changes. Hence, a lid that allows performing such tasks was designed and 3D printed.

This lid should allow the assembling of plastic tubes to guide the fluids and keeping the culture µ-Dish as sealed as possible at the same time to avoid contamination events. Further, it should also enable taking images without any kind of interference. Hence, the dimensions of the originals µ-Dish and lid were measured with a digital calliper. The selected µ-dish has 35 mm total diameter and a cell growth area diameter of 21 mm. This way, the final lid was designed taking into account that it should not be covered the inner cell growth area, since commonly available and cheap 3D materials are opaque and it would not let to perform a viable image acquisition.

3.1.2. Software

In order to reach the purposed goal, significant changes have been performed to set Micro-manager functional to communicate with all the pumps connected. The communication between the open-source microscopy software and the Arduino is performed through a serial port, where the Arduino is waiting to decode the information and translate it in pump tasks. The Micro-manager complete source code was downloaded with a subversion client. The SLIKSVN was the client installed and then all the source code was checked out, including the Java and C++ software architecture layers. Other required tools was installed for building and debugging Micro-Manager source code such as Microsoft Visual Studio 2010 SP1 Express for the C++ components, Oracle Java Development Kit and NetBeans for the Java code components, as suggested at Micro-manager website. Finally, the build tool Apache Ant was also installed to automate the full Windows build of Micro-Manager [34].

3.1.2.1. Arduino Sketch

Micro-manager has already developed device drivers to control Arduino, commanding it to perform a limited range of tasks. The Arduino firmware (sketch) available on Micro-manager website allows the Arduino behaving as a HUB device, which can read a switch, a state, an analog input, connected device, among other features. However, this firmware was not able to command a stepper motor. So, changes have been made in order to correctly control the stepper motors that drive the syringe pumps.[35]

The sketch is organized basically as a "*switch case*" loop. A serial port communication is initialized and then the Arduino waits for an available byte on it to read. This byte gives information about the task Arduino should perform. Hence, the byte content is evaluated by the switch case loop testing all the task capabilities. After selecting the correct case branch, the Arduino can receive additional bytes from the serial port to properly perform the demanded task.

Pololu has already an Arduino library available to download, for an easy control of some of their stepper motor drivers, including the DRV8834. This library was added to the Arduino software and then it was analysed. It is a complete library that allows doing all kind of stepper motor tasks, including changing the microsteps resolution, setting the angular speed of the stepper motor (in revolutions per minute), and performing an accurate rotation of a certain number of steps or degrees [33].

So the main changes applied to this firmware consisted in including this library and adding three DRV8834 stepper motor driver class objects, one for each stepper motor. Then, two new cases were also added to the pre-existing switch case loop: one for calibration tasks and the other to run a stepper motor a certain number of steps, in one direction and at a certain speed. Furthermore, the version of this firmware was changed from 2 to 3 to resolve possible disambiguation errors. The new firmware version can be seen at Appendix D.

3.1.2.2. C++ source code

Micro-manager device adapters are written in C++. The Microsoft Visual Studio 2010 SP1 Express was used to load this source code component, as suggested in the tutorial available at Micro-manager website. The Visual Studio solution was organized in several projects, each one for a device adapter. Since the main purpose was only editing the Arduino device adapter, its Visual Studio project was isolated and added to a new empty Visual Studio solution. Furthermore, other projects had to be added due to compiling dependencies [34].

As described in the Micro-manager “Building Micro-Manager Device Adapters” tutorial, the Arduino behaves as a HUB device, i.e., the Arduino may not perform any concrete task but it is crucial to manage and send commands to control other devices that are attached to it, called “child” devices (Figure 35) [36].

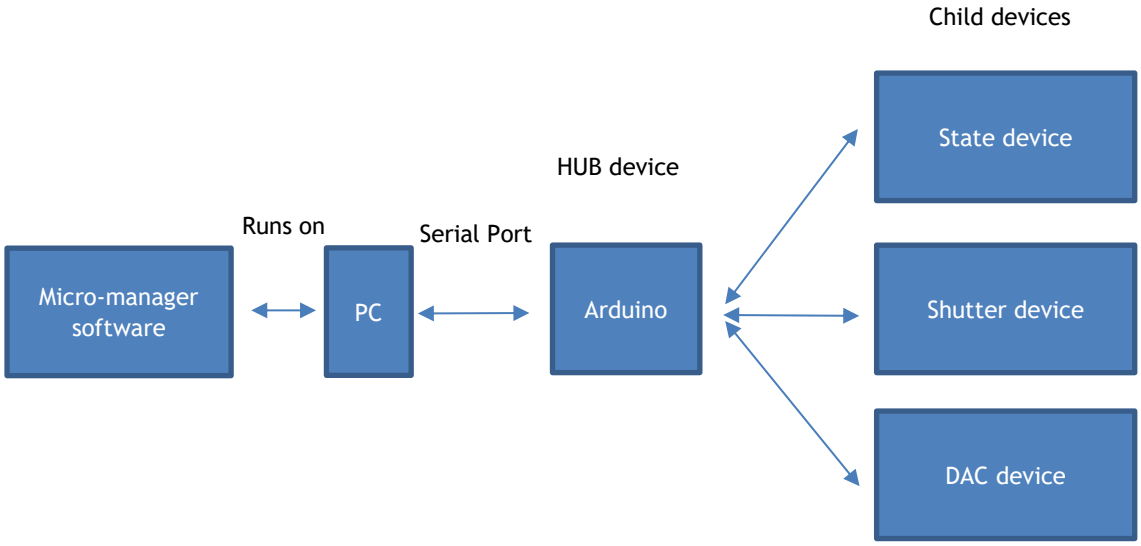


Figure 35 - Micro-manager and Arduino communication scheme.

First of all, the CArduinoHub class that controls the initial overall communication with the Arduino through a COM serial port is declared, and then it detects the devices that can be controlled by the Arduino.

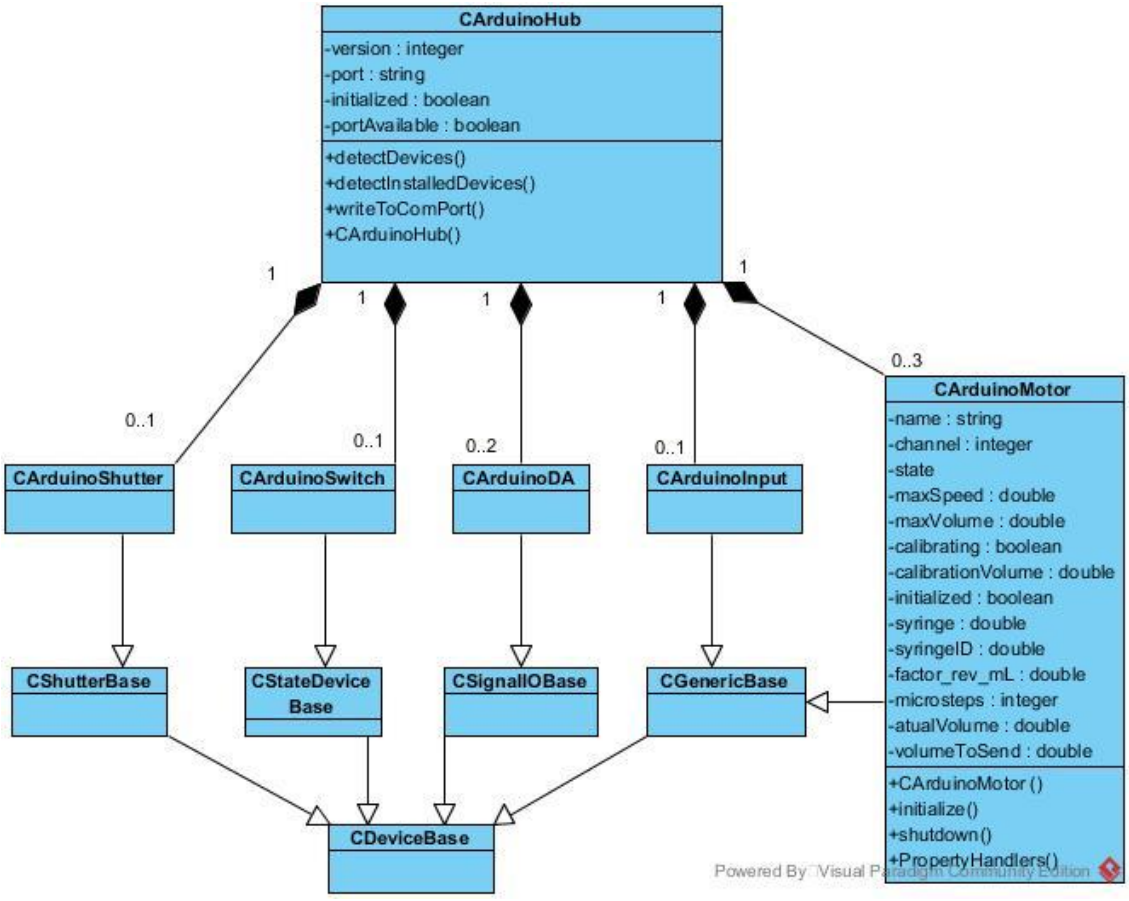


Figure 36 - Device Adapter class diagram

Further, a range of C++ classes are declared, one for each child device, in order to control specific characteristic parameters depending on its type. Hence, a new class was created in order to control a new type of child device, a stepper motor. This class was labelled as a generic device type class and public and private methods were also created. Following the device adapter methods suggested by Micro-manager tutorial, a constructor and destructor are created and then initialization and shutdown methods are also defined. Then, a range of properties had to be created. These properties are features that characterize the child device and that can be modified or not by the final user when using the Micro-manager software. In this case, the properties created are described in Table 11.

Further, a range of property handler methods was created in order to capture property value changes during the operation, one for each editable property. Following the code style of the property handler methods in the other child device classes, the property handler method should begin with an "On" prefix, followed by the property attribution name. Each property handler is normally composed by a set and get method that control and modify the property value when necessary. Furthermore, some of them can also trigger a COM port communication with the Arduino.

Table 10 - Property handler methods.

Property handler method	Property being controlled	Additional information
OnActualVolume	actualVolume	
OnVolume	volume	Triggers data sending to the Arduino in order to perform infuse/withdraw of a certain amount of volume
OnWell	well	
OnCalibration	calibration	Triggers data sending to the Arduino in order to perform syringe volume calibration
OnCalibrationVolume	calibrationVolume	

A private method, designated as WriteToPort, was created in order to manage the COM serial port communication. This method accepts as input the action that should be sent to the Arduino and depends on where this method is called. When OnCalibration property handler detects a change in the calibration property, the WriteToPort method is called with an integer action input 51. On the other hand, when the OnVolume detects a change in the volume property, the WriteToPort method is called with an integer action input 50.

The WriteToPort method creates a byte array that is stored as a sequence of bytes that will be sent to the COM serial port, in order to be read by the Arduino. Depending on the input action, this array contains different information.

The first byte to be sent corresponds to the input variable. This byte is also the first byte that will be read and analysed by the switch case loop. Thus, it is important that this first byte be unique in order that the Arduino clearly understand which task it should be performed and how many bytes it should read more.

Table 11 - List of properties created.

Property	Label	Object type	Read only	Allowed values	Property description
Description	"Description"	String	yes	"Arduino motor driver"	Defines a general description of the child device
Name	"Name"	String	yes	"Arduino-Motor1"	Creates a different label for each stepper motor
				"Arduino-Motor2"	
				"Arduino-Motor3"	
Syringe	"Syringe (mL)"	Integer	yes but pre-defined by user before initialize method	1	Defines which syringe is being used
				5	
				10	
				20	
				50	
maxSpeed	"MaxSpeed (rpm)"	Float	yes	200	Defines the maximum speed allowed
actualVolume	"Volume available (uL)"	Float	yes	Unsigned float	Store the current volume contained on syringe
Volume	"Volume (uL)"	Float	no	Signed float	Stores the volume delivered
Well	"Well"	Integer	no	Ranging between 1 and the number of culture chamber wells defined	Stores the well label where the volume should be delivered
calibration	"Calibrate"	String	No	"On"	Defines if the pump is calibrating or not
				"Off"	
calibrationVolume	"Calibration Volume (mL)"	Float	No	Unsigned float	Stores how much volume will the syringe be calibrated to

To perform infusion/withdrawing of a certain amount of volume, the first byte of this array is 50. In this case, 6 more bytes are needed in order to perform a correct pump task. The second byte gives information about the pump that must perform the pump task. The third byte stores information about the speed at which this pump task must be performed. The following four bytes give details about the volume that must be infused/withdrawn. The fourth byte corresponds to the direction for which the motor shaft must rotate. When the volume value is negative, the specified amount of volume will be withdrawn. On the other hand, when the volume value is positive, the specified volume will be infused. The remaining three bytes correspond to the volume that should be properly sent. The volume inserted by the user is converted to a microsteps resolution. Hence, a revolution per volume unit factor must be calculated and this factor is dependent on which syringe is being used. In order to calculate these factors, an initial calibration method was defined and it will be explained in section 3.1.3. Since the integer converted microsteps result can get high values, three bytes were needed to avoid information to be lost. Hence, the result was converted to binary and then decomposed in the most significant byte, intermediate significant byte, and least significant byte.

To perform a calibration step, the first byte of this array is 51. In this case, only two more bytes are sent, one to give information about the pump to be calibrated and the following byte to send the integer millilitre volume at which the pump must be calibrated.

The actualVolume property value is changed when one of these actions are called. When the calibration step action is called, the actualVolume property is set to the calibration volume that has been sent. When performing a pump task, the actualVolume property is readjusted depending on volume and direction that has been sent. This is a fundamental procedure since before performing infusion or withdraw pump tasks, the system checks if the syringe has sufficient remaining fluid volume to infuse or if there is enough space to withdraw the desired volume, respectively, in order to avoid damages.

The Arduino device driver project was then build using x64 configuration, which corresponds to the Micro-manager version installed. The DLL file was then copied and replaced the pre-existing Arduino device driver DLL file at Micro-manager installed root.

3.1.2.3. Java source code

After compiling the C++ source code, the pumps could be already controlled by the Micro-Manager software using the Script Panel. After performing the Hardware configuration and including the Arduino HUB and the selected devices, all the related properties must appear at Device Property Browser.

However, the main final users of this solution are biologists, reason why the developed script environment can be seen as a big barrier. Hence, a more user friendly GUI solution was required.

Micro-manager has a powerful tool called Multi-Dimensional Acquisition that can perform a well-defined image acquisition. This image acquisition is performed along several dimensions, from time interval to a range stage positions. Hence, it was thought that a new tool could be added to this acquisition engine, consisting in the capability to define a list of pump task that can be triggered when performing a Multi-Dimensional Acquisition protocol.

To achieve this goal, it was defined that the best way to fulfil all the requirements was to create something similar to the multiple XY positions GUI. As seen in Figure 37, when opening the Multi-Dimensional Acquisition tool, a new window pops out and here, all the dimensions can be defined before starting image acquire. However, in certain cases the input data that must be inserted may not functionally fit the windows size. Hence, in certain cases, as the Multiple Positions (XY) and Autofocus checkbox pane, a button can be pressed in order to open a new window to insert this input data. The user selects the checkboxes in the Multi-Dimensional Acquisition window, to select if it is desired to use this new tool during the current acquisition.

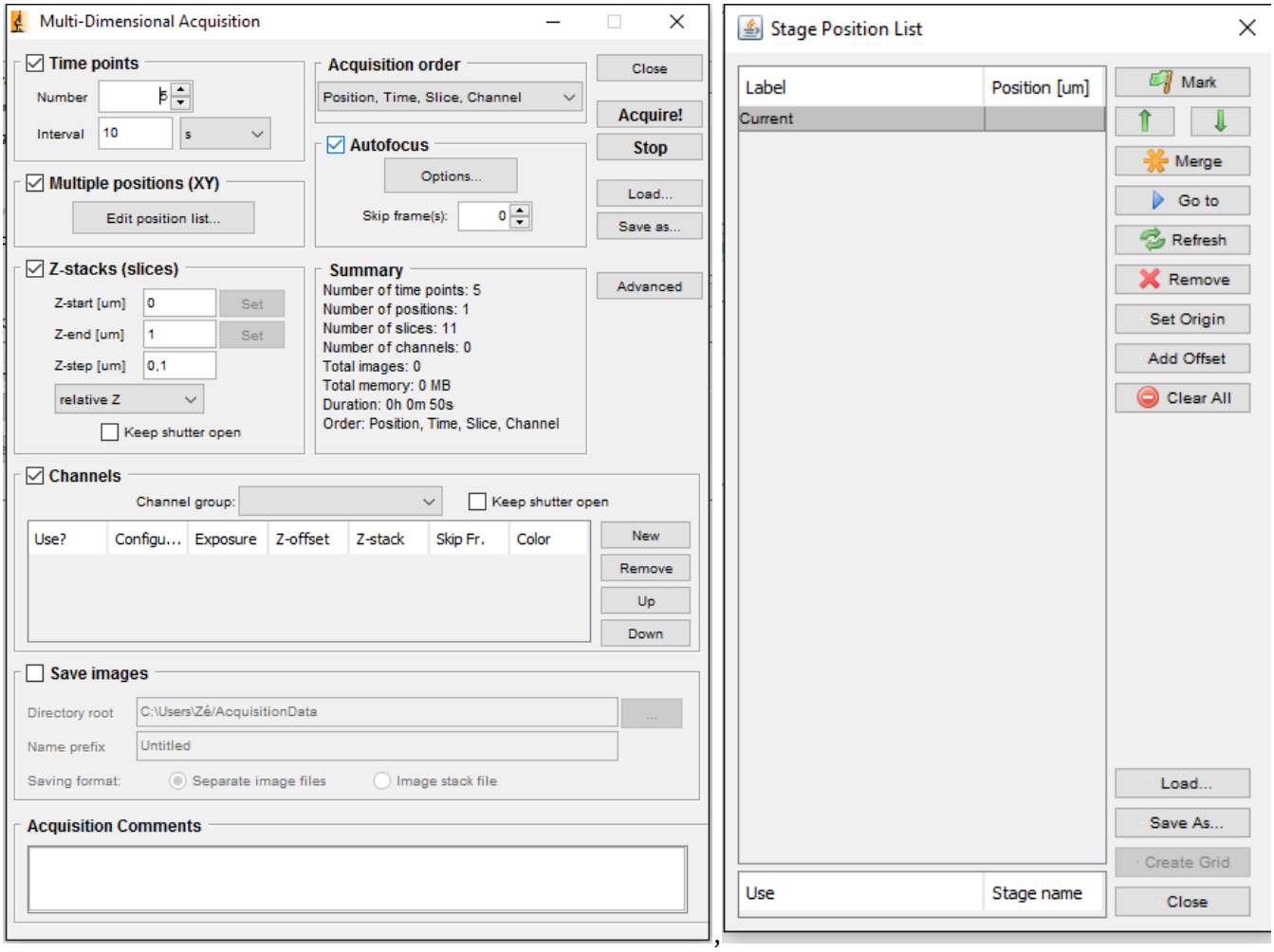


Figure 37 - Multi-Dimensional Acquisition and Stage Position List GUIs.

For instance, when selecting the “Edit position list” button, a new window pops out and here the stage positions that must be covered are inserted. As mentioned before, a similar way of working was adopted in order to create the pump task GUI, since a lot of information must be inserted by the user regarding a pump task sequence. So, a new checkbox pane was added to the Multi-dimensional Acquisition window labelled as “Multiple pump-time tasks”. Here a button was added and when pressed a new windows should appear in order to input the desired pump task sequence. However, these pump tasks should be only performed when the checkbox is selected. When deselect and even if it was inserted a pump task sequence, this data must be ignored.

A new window dialog pops out when the button is selected. The easiest way to insert the pump task sequence is adding values to a table. This new window should show a table with some fixed columns and then the user must have the capability to insert new rows and edit its cells content.

The first input data that the user must insert is the time which a certain pump task should be triggered. Thus, the first column corresponds to information about the time variable.

Sometimes, the user might want to perform a repetitive task as a medium change task. Hence, when adding a new medium change to the pump task sequence, the user should have to proceed to the unpleasant task of inserting repetitive volumes. It was thought that it should be valuable to create a branch of predefined pump task types since inputting repetitive volumes for repetitive tasks might be boring and time useless. Three pump task types were defined from the requirements collected: simple volumes deliver, medium change and syringe calibration. A column corresponding to a task type was added.

Posteriorly, three more columns were added, one for each pump. These columns should save particular information for each pump and its content may vary depending on the pump task type selected. For a simple volume deliver task, each column should show an editable number field so the user can insert the volume desired, positive for an infuse task and negative for a withdraw task. For a calibration task, an editable checkbox should appear on each column. Here the user can select the pump(s) that must be calibrated at a certain time. When

performing this kind of task, the user must insert the fluid volume contained on syringe which can be changed using push-buttons. On the other hand, for a medium change pump task, these three columns shouldn't be editable at all, but the information regarding the medium change procedure desired should be inserted. Hence, auxiliary text fields and combo boxes were added in order to insert the medium change infusion and withdraw volumes, and select which pump will infuse and which pump will withdraw, respectively. However, each one of these three columns should only be editable if the respective pump is already configured.

Furthermore, another column was added to represent the sequence that should be followed when performing a simple volume deliver or a calibration pump task. The best way is to select the desired order for a certain pump task from a combo box, where it should be available all possible permutations between the three pumps.

Table 12 - Pump task table column details.

Column index	Column Name
0	Time
1	Task Type
2	Pump1
3	Pump2
4	Pump3
5	Order

Additional buttons were inserted in order to manage the table content as an add and a remove button, to add a new pump task instance to the pump task sequence or remove a selected one, respectively. Furthermore, a calibrate button should was also inserted in order to perform an initial calibration. When this button is selected the user should be requested to insert the volume at which each syringe should be calibrated. Another useful button that was created is the merge and order button. When selected, the pump task sequence must be ordered by time instance, and multiple pump tasks of the same type performed at the same time should be merged at a single pump task. This procedure must also be performed just before saving the pump task sequence, when the user commands beginning the acquiring process, in order to generate a more efficient and organized output data.

Then, it was thought that a valuable button could be added, responsible for triggering the running of the pump tasks sequence out of the Multi-Dimensional Acquisition process. This would be useful to test if the pump task sequence was well defined, or just to run it without image acquisition.

3.1.2.4. Clojure source code

After analysing the java source code, when clicking the Acquire button at the Multi-Dimensional Acquisition window, the sequence settings are saved and then transferred as input of a clojure written method. As proceeded with the position list object for sending the multiple position targets, the pump task list was also sent to this clojure written method. However, the handling that is performed by the source code to these input parameters should be different. In the case of the multi positions list, the algorithm combines this dimension with other dimensions that are related with image acquisition instances, as time points, z-stacks (slices) and the selected channels following the acquisition order selected. On the other hand, it was decided that each pump task instance must not trigger any kind of image acquisition.

After generating the image acquisition sequence, each image acquisition instance is triggered when the correspondent elapsed time is reached. Hence, the best way to perform the pump task sequence is to intermediate the image acquisition sequence directly with the pump task list since it already comes ordered. Besides being tested if it is time to perform an image acquisition, it was added an "if" clause that also checks if it is time to perform the next task.

3.1.3. Syringe pump revolution per volume unit factor

When performing a specific pump task, the user should define the volume to be infused/withdrawn by the syringe pump. The stepper motor driver controls the stepper motor shaft rotation, which in turn drives the

syringe plunger linear movement. When the stepper motor movement is requested, the number of steps to be performed should be sent to the stepper motor driver. Hence, a calibration factor should be measured in order to convert the volume inserted by the user into a number of steps that must be performed by the stepper motor. However, this factor depends on which syringe is intended to be used. Thus, a total of 5 revolutions per volume unit factor were measured, one for each syringe size used.

An Arduino sketch was uploaded and one of the buttons that were added to the electrical circuit was used. When clicked, the stepper motor driver commands the stepper motor to rotate one revolution.

After assembling a certain syringe containing water with a Luer lock needle at the end to the syringe pump, the button was pressed and the volume was collected to an Eppendorf 1.5 ml tube. The tube was sealed and its mass was measured using an analytical balance.

The mass of the volume contained in the tube was calculated, through the difference between the measured mass of the tube plus the volume contained and the measured mass of the empty tube. The volume contained was then estimated from the measured water density value and the revolution per millilitre factor was then calculated using the following equation:

$$Rev\ per\ mL\ factor = \frac{n_{revolution}}{\frac{m_{eppendorf+fluid\ volume} - m_{empty\ eppendorf}}{\rho}} \quad (3.4)$$

Where $m_{eppendorf+fluid\ volume}$ represents the measured mass of the Tube after the fluid injection, in gram, $m_{empty\ eppendorf}$ the measured mass of the Tube before the fluid injection, also in gram, and ρ the water density in g/mL. When performing only one revolution, the Tube volume capability was not well used since only a small amount of volume was injected. Thus, it was done more than one revolution for each assay in order to better fill the Tube's volume capability. Instead of performing only one revolution for the 1 mL, 5 mL, 10 mL, 20 mL and 50 mL syringes, 10, 5, 4, 3 and 2 revolutions were produced, represented by $n_{revolution}$ in equation 3.4. Five assays were performed to ensure repeatability.

Due to the small amounts of values that were being calculated, it was evaluated the reliability of the obtained results can be affected.

The plunger position is linearly related with the number of revolutions performed:

$$n\ revolutions\ \propto \Delta\ plunger\ position \quad (4.1)$$

Assuming that syringe body is a cylinder with linear inner diameter:

$$Volume_{cylinder} = radius^2 \times \pi \times length \quad (4.2)$$

$$\Delta\ Volume_{cylinder} = radius^2 \times \pi \times \Delta\ length \quad (4.3)$$

$$\Delta\ length = \frac{\Delta\ Volume_{cylinder}}{radius^2 \times \pi} \quad (4.4)$$

$$n\ revolutions\ \propto \frac{\Delta\ Volume_{cylinder}}{radius^2 \times \pi} \quad (4.5)$$

$$\frac{n\ revolutions}{\Delta\ Volume_{cylinder}}\ \propto \frac{1}{radius^2 \times \pi} \quad (4.6)$$

It can be said that the number of revolutions per volume unit factor is linearly proportional to the inverse square radius. Hence, it was calculated the slope of this regression using the revolution per unit factor estimated before and the inner diameter of each syringe body measured with a digital calliper. This would let the system support every type of syringe ranging from 1 mL to 50 mL inputting only the inner diameter of the syringe that is being used.

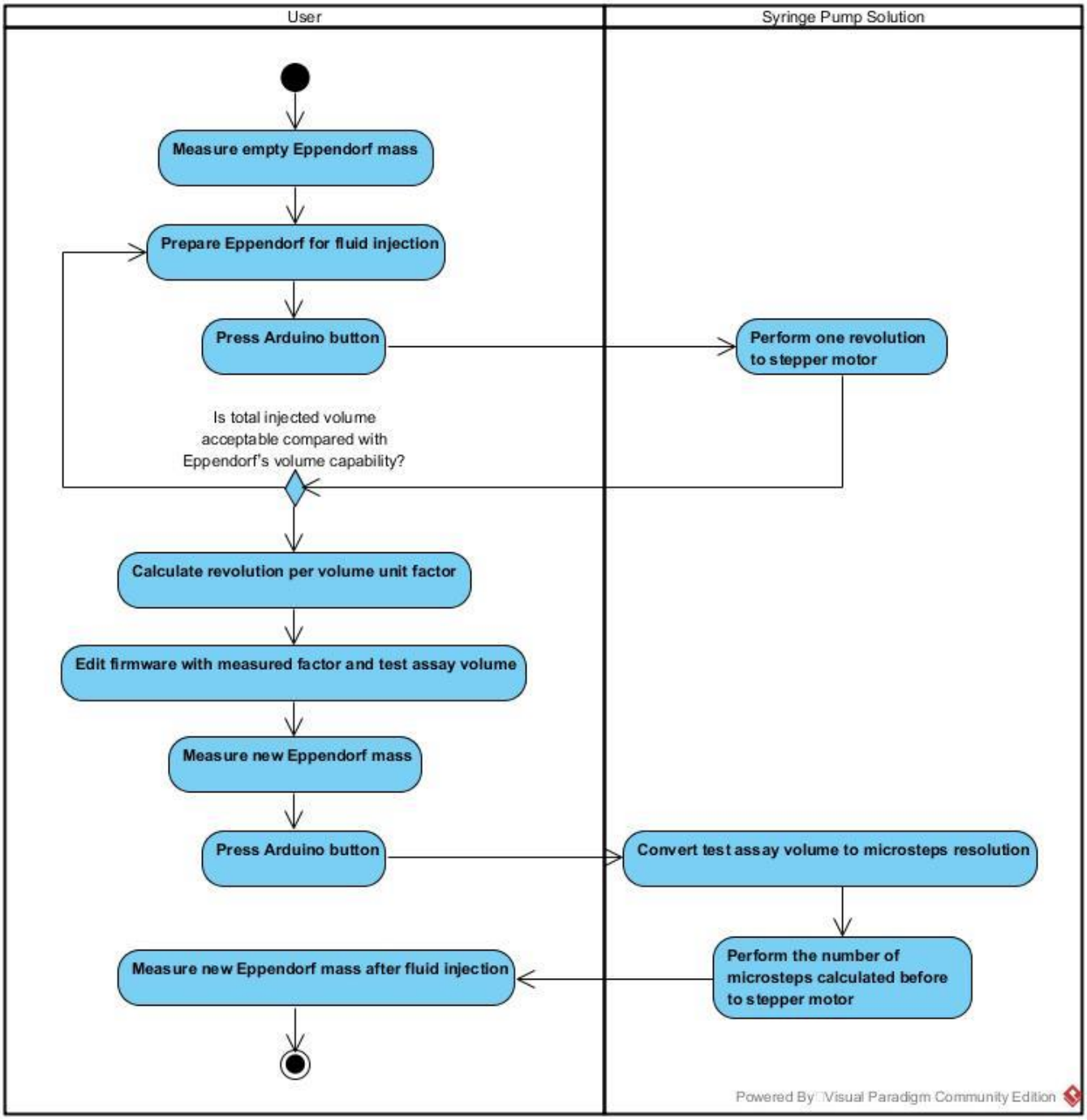


Figure 38 - Activity Diagram

3.2. Proof of Concept

After correctly assembling the system, there was a need to verify if the designed solution was suitable to use when performing live cell imaging protocols. Thus, two experiments were performed in order to verify the two main mechanisms that the solution can accomplish: culture medium replacement and simple amount of volume delivery, at a specific time point.

The first protocol aimed to verify if the medium replacement procedure was well implemented. The cells can undergo by a natural division process designated mitosis, which comprises several phases. There are some drugs that can be added to the culture medium in order to avoid cells of suffering mitosis. However, when performing culture medium replacement with fresh medium without drug, the cells can resume their natural mitosis cycle. So, the first protocol was designed in order to verify the event above mentioned. Thus, fibroblasts were cultured at an IBIDI 35 mm diameter µ-Dish with S-Trityl-L-cysteine (STLC) drug, which block cells at anaphase. The designed solution was used to perform culture medium replacement, in order to remove the drug in the culture. To validate that cells resume mitosis process from the blocking point and certify that the mitosis is completed, a time-lapse microscopy movie was recorded.

The second protocol simulates the infusing of a drug in a cell culture at a specific time point. Fibroblasts were cultured. In order to make addition of the drug visible, it was used a fluorescence marker specific to stain mitochondria inside living cells (MitoTracker® Deep Red FM, Thermo Fisher Scientific). A time-lapse microscopy image using the MDA tool of µ-Manager was recorded. The system was set to infuse the marker 5 minutes after the initial time-lapse image acquisition.

For both proof of concept experiments, the 35 mm µ-dish designed lid was attached to the culture dish. In the first protocol, two syringe pumps were calibrated and used: one to withdraw the medium in culture and one to infuse fresh medium. On the other hand, in the second experiment, only one syringe pump was needed to perform the drug infusing.

The microscope used was Zeiss Axiovert 200 M (Carl Zeiss) with a camera CCD CoolSnap HQ (Roper) and a Prior XY motorized stage (PRIOR) all controled by Micro-manager software. The microscope was enclosed into an environmental chamber (OKO-LAB) the keeps the cells at 37º C and 5 % CO_2 while performing the time-lapse imaging.

Figure 39 - Zeiss Axiovert 200 M

Chapter 4

Results and Discussion

4.1. MOST syringe pump modifications

The modifications mentioned before were performed and the 3D printed components were printed.

Firstly, the clamp was modified in order to design two different types: one to hold up to 10 mL syringes (Type 1) and the other up to 50 mL syringes (Type 2). Thus, their diameters measure 174 mm and 313 mm, which are the 10 mL and 50 mL syringe body outer diameter, respectively.

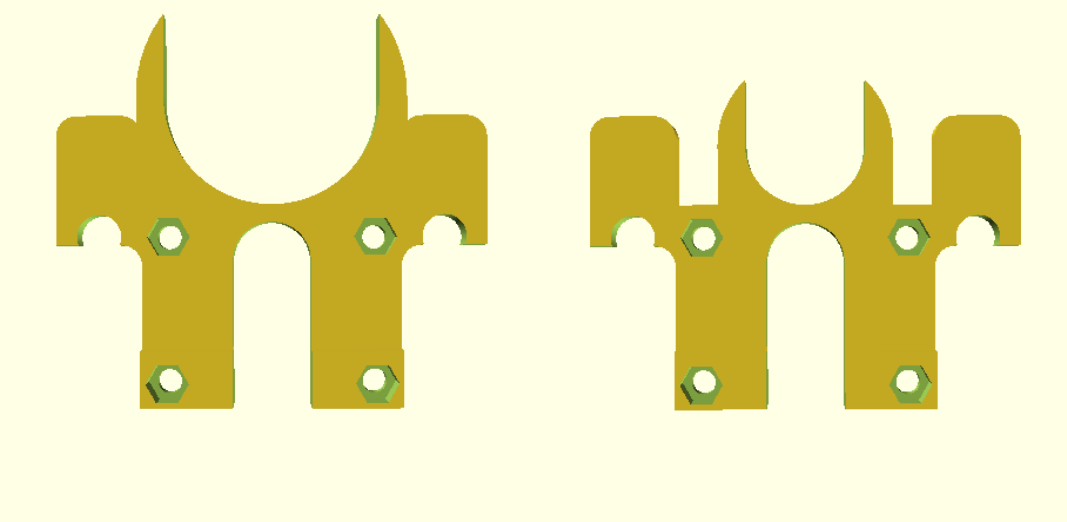
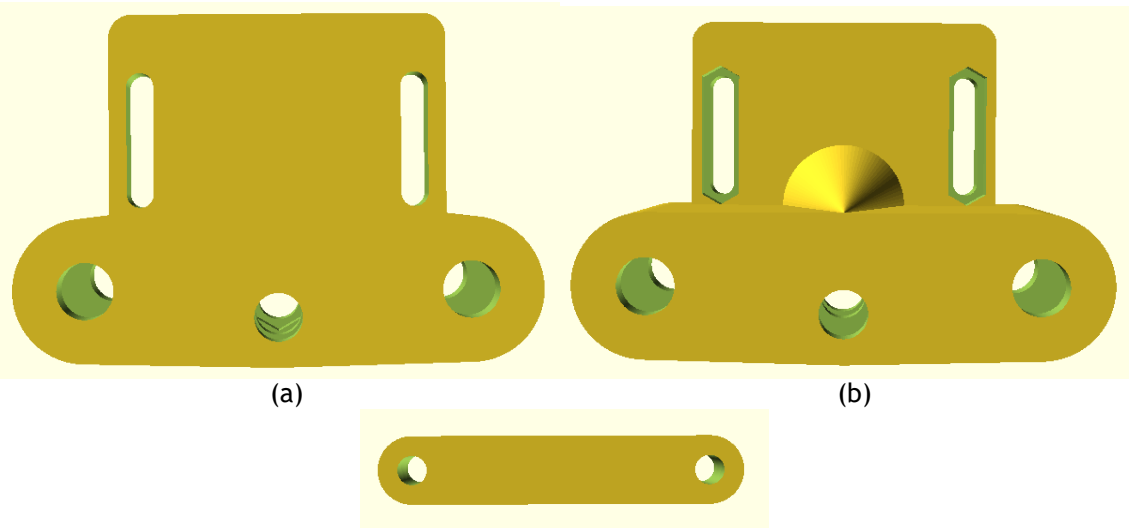


Figure 40 - Clamp types (a) type 1 - diameter 17,4 mm (b) type 2 - diameter 31.3 mm (tolerance fit included).

Then, the carriage was adapted to secure the plunger, having into the account the syringe size partition mentioned before.

Initially, two horizontal bars were designed to hold the syringe plunger (Figure 41).

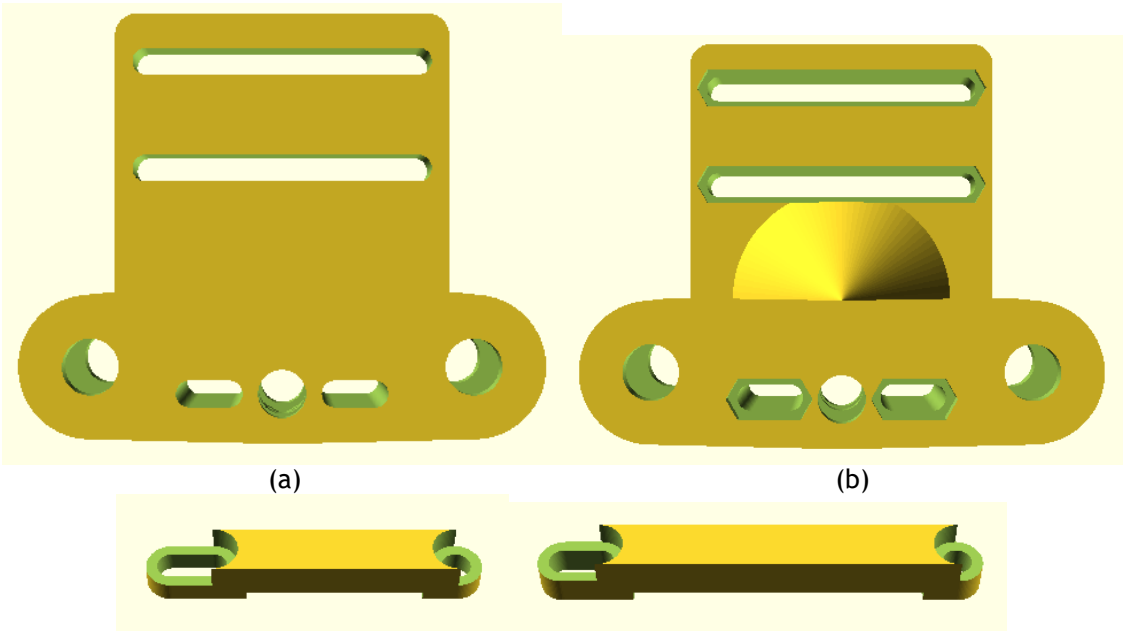


(a) (b)

(c)

Figure 41 - Initial carriage and plunger retainer design: (a) Carriage front view (b) Carriage back view (c) Retainer

However, this configuration was not functional, because there was not enough space for the bottom bar to properly retain the syringe plunger. Hence, another approach was successfully performed using vertical bars instead of horizontal ones.



(a) (b)

(c) (d)

Figure 42 - Final carriage and plunger retainer design: (a) Carriage front view (b) Carriage back view (c) Retainer type 1 (d) Retainer type 2

Two parallel horizontal rips were added on top of the carriage to fix the top of the retainer, one for each plunger retainer type, and two symmetric horizontal rips for the bottom fix. The plunger retainer was also designed with a gap under it, in order to tolerate the plunger thickness, and at the screw insertion point (edges) a lower thickness was applied, in order to allow more linear movement range to the carriage. It is important to use a higher useful volume of the syringe. The same reason also leads to change the thickness of the clamps and the end idler.

Table 13 presents a list with 3D printed components and quantities needed to print for each syringe pump unit and Figure 43 shows these listed components after being printed and before assembling process.

Table 13 - List of 3D components for one syringe pump.

3-D Printed	Count
Motor End	1
Carriage	1
Retainer – Type 1	2
Retainer – Type 2	2
Clamp – Type 1	2
Clamp – Type 2	2
Idler End	1

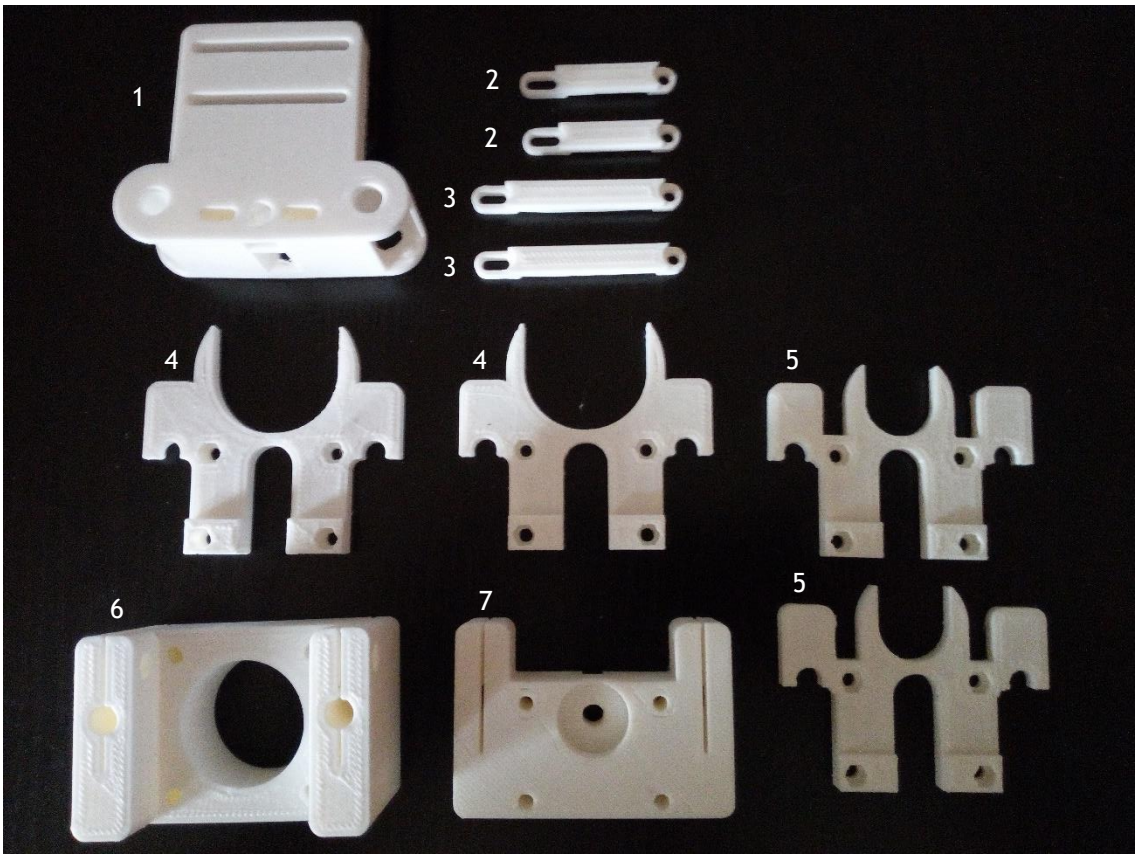


Figure 43 - 3D printed components: (1) Carriage (2) Retainer – type 1 (3) Retainer – type 2 (4) Clamp – type 2 (5) Clamp – type 1 (6) Motor end (7) Idler end

The pump was assembled as suggested in MOST wiki page. Firstly, the stepper motor was fixed with M3 x 20mm screws to the motor end 3D printed component. Two 20 cm metal rods where laterally attached to the motor end and the 20 cm threaded rod was coupled to the 5 mm diameter stepper motor shaft. Then, to assemble the carriage component, two LM6UU linear bearings were inserted to its lateral parts and a M5 trap nut was inserted in a proper hole under it.

The user must choose which syringe type he wants to use since retainer and clamp components combined in each assembling configuration should be from the same type

After selecting it, the retainer should be fixed to the carriage component. Each retainer bottom end is fixed to one of the carriage bottom horizontal rips. The retainer top end is fixed to the carriage top rips depending which retainer type is being used. For a Type 1 retainer, the middle horizontal bar should be used. For a Type 2 retainer, the highest one should be the one selected. The two possible carriage-retainer configurations are represented in Figure 44.

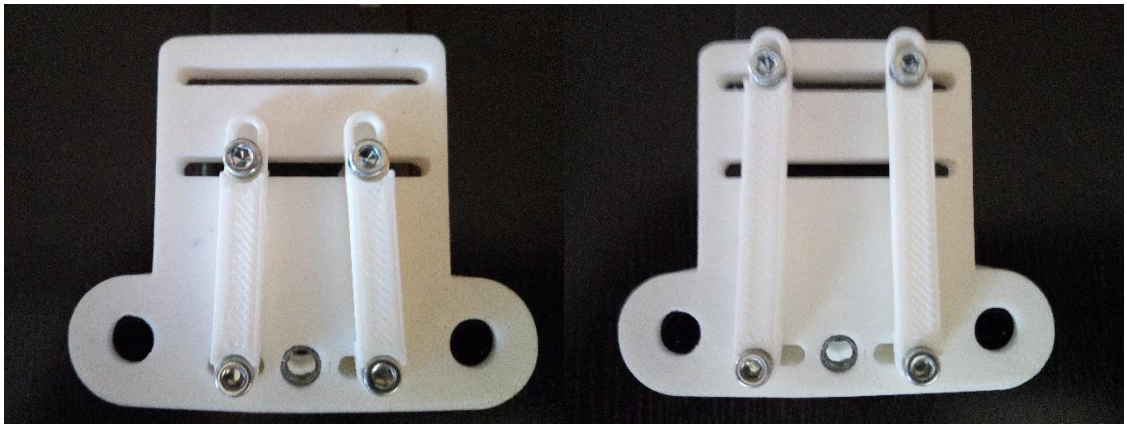


Figure 44 - Possible retainer assembling to carriage (type1 and type2)

As observed, both horizontal bars were ripped, in order to fix the new retainer that has been designed allowing the plunger syringe to follow the carriage movement.

A M3 screw and nut couple fix each retainer end point to the carriage bar. The two top horizontal bars rip the carriage surface in its possible maximum length to promote an easy syringe removal procedure. The two symmetric lower bars allow less range of movement freedom due to geometric limitations.

Two 625z ball bearings are trapped into the idler end component. Then, two clamps are fixed with four M3x35 mm screw and M3 nuts to the idler end. Lastly, the idler end is mounted on the lateral rods.

The selected syringe body is fixed with the finger flange between the two clamp components and its plunger should touch the carriage surface. Both retainer top end positions are adjusted in order to place them as close as possible to correctly secure the syringe plunger.

The final aspect of the assembled pump can be observed in Figure 45, and the complete material list is listed below.

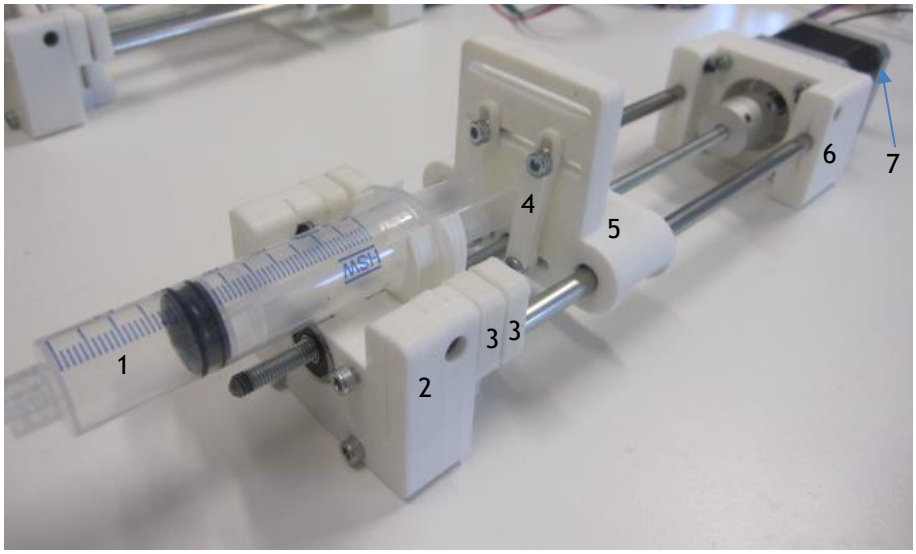


Figure 45 - Syringe pump assembled: 1 - Syringe; 2 - Idler end; 3 - Clamp; 4 - Plunger retainer; 5 - Carriage; 6 - Motor end; 7 – Stepper motor

Table 14 - Required material to assemble one syringe pump.

Materials	
3-D Printed	**Count**
Motor End	1
Carriage	1
Retainer – Type 1	2
Retainer – Type 2	2
Clamp – Type 1	2
Clamp – Type 2	2
Idler End	1
Motors & Metal	**Count**
NEMA17 motor	1
5mm x 5mm shaft coupling	1
625z ball bearing	2
LM6UU linear bearing	2
M3 x 10mm socket head cap screw	4
M3 x 20mm socket head cap screws	6
M3 x 35mm socket head cap screws	6
M3 hex nut	16
M5 hex nut	1
M5 threaded rod 0,2 m	1
6mm A2 tool steel 0,2 m	2

After assembling all three pumps, the electronic circuit was mounted on a breadboard. The required pin connections were performed as listed in section 3.1.1.3 except the stepper motor wiring. Before that, the stepper motor driver current limitation was set to 1 A per phase. With the external 9 V power supply disconnected, the stepper motor wires were connected to the correspondent driver pin. The final aspect is shown in Figure 46.

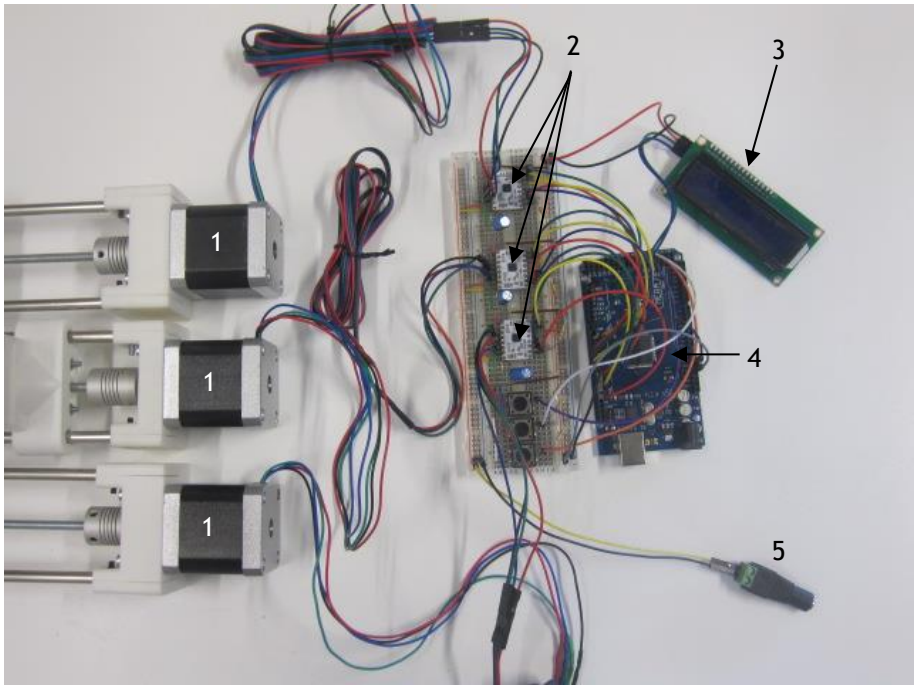


Figure 46 - Final prototype: 1 – Syringe pump; 2 – Stepper motor driver; 3 - LCD; 4 – Arduino board; 5 – Power supply input

A short Arduino sketch was uploaded to check if all pumps were totally functional. After confirmed, the revolutions per millilitre factors were estimated.

In the begging of this project, it was stated that the solution built should be a low-cost solution. The costs associated are only related with hardware acquisition, namely electronic components, 3D-printed parts and metal components.

At Appendix C, it can be found a detailed table that describes the costs of each component. Each syringe pumps had a total cost of 50,76 € (3D-printed components, stepper motor and fitting metals) and the final functional system with three syringe pumps had a total cost of 213,22 € (including microcontroller, stepper motor drivers, and 3D printed µ-dish lid).

Besides being important, the final cost may not be seen as the only important factor. Another aspects are important too as the functional capabilities of each product. Pearce et al. had already reported a comparison table between their syringe pumps solutions and commercial ones.[28]

The syringe pump built cost more or less the same compared with MOST solutions and it stills a much cheaper solution compared with commercial ones. Moreover, some of these commercial syringe pumps are more expensive than the open-source ones and present weak functional capability as infusion capability only or worse assessment results.

4.2. Revolutions per millilitre factor

Before starting the tests for each syringe, it was studied which kind of variables could affect the calibration procedure, in order to minimize possible errors.

The first variable included was the density of the water that was being used. It was tested if a water density estimation of 1 g/cm3 would be reasonable. Hence, the water density was indirectly measured inserting 400 µL into an Eppendorf 1.5 ml tube using a micro-pipette. The mass of the fluid was measured and the water density was obtained. Five assays were performed as described in Table 15 and all weighing were performed with an analytical balance.

Table 15 - Water density measurement results.

Assay	Empty Tube (g)	Tube +100 µL of water (g)	100 µL of water (g)
1	0.93882	1.33813	0.39931
2	0.93483	1.33653	0.40170
3	0.94709	1.34842	0.40133
4	0.94715	1.34823	0.40108
5	0.93700	1.33795	0.40095
		Average	0.40087
		Standard deviation	0.00063
		ρ	1.00219

It was obtained a water density of obtained 1.00219 g/cm3. A small difference was observed compared with the initial water density estimation of 1 g/cm3. Besides being small, this difference can introduce higher deviations when calculating the revolution per mL factor, due to the small values that its intermediates calculations take. Hence, the average of the water density values measured was considered for future calculations.

Then, it was tested if there was a significant variability between empty Eppendorf tube masses. Thus, the mass of 36 empty tubes were measured and it was obtained an average value of 0.94547 ± 0.00427 g (0.45% of the average). When converted to volume units, with the average water density estimated before, the volumes deviations can reach values higher than 10 µL, which would not be considered reliable for this project purposes. Hence, instead of calculating the mass difference with the average empty tube mass, each tube was labelled and the mass difference was calculated for each one.

The needle type attached at the end of the syringe seems to affect the results variability. Blunt and bevel needle are the most used types in experimental protocols. These needle types were tested in order to use the one that would bring less variability between results.

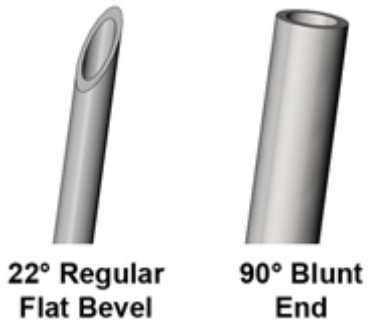


Figure 47 - Bevel and blunt needle types [37]

It was verified that the bevel ones accumulates on its tip higher drops of water after a volume injection since they have a higher contact area at its tip. It was also observed higher water drop dimension variability between sequences of equal volume injections than the observed when using blunt needles. These two reasons

lead to discard this type of needle, since it could bring more errors and variability to this calibration step results. A 19G x 1'' blunt needle was the needle selected.

Finally, it was studied the needle end position when performing the fluid injection. The variability between volume results was tested when the fluid injection is done with the needle tip leaning or not leaning the Tube inside wall. So, three assays were performed for each needle end position defined: leaning or not leaning the Tube inside wall. The results are presented in Table 16 and Table 17

Table 16 - Needle tip not leaning the Tube inside wall results.

Assay	Empty Tube (g)	Tube + volume injected after 10 revolutions (g)	Volume of 10 revolutions (g)
1	0.94232	1.07279	0.13047
2	0.95411	1.10764	0.15353
3	0.93954	1.08581	0.14627
		Average (g)	0.14342
		Sd (g)	0.00864
		Sd (%)	6.02103

Table 17 - Needle tip leaning the Tube inside wall results.

Assay	Empty Tube (g)	Tube + volume injected after 10 revolutions (g)	Volume of 10 revolutions (g)
1	0.95021	1.10027	0.15006
2	0.94771	1.0887	0.14099
3	0.94562	1.09103	0.14541
		Average (g)	0.14549
		Sd (g)	0.00305
		Sd (%)	2.09565

When performing the fluid injection with the needle tip not leaning the Tube wall, a higher standard deviation was obtained between the three samples results. This occurs because after performing a volume injection, the drop volume retained at the needle can vary greatly. The initial drop volume variability, conjugated with the drop volume variability after a fluid injection, may lead to a higher error propagation. The results support this conclusion, since the standard deviation obtained were 6.02 % and 2.10 % when performing fluid injection with needle end not leaning and leaning the Tube inside wall, respectively. Hence, to reduce this variability, the water injection was performed with the blunt needle end contacting the Tube wall.

After all these tests, the revolutions per volume unit factors were estimated. The tests were performed using 1.00219 g/mL of water density estimation, a blunt needle and the needle end leaned on the Tube inside wall. The stepper driver microsteps resolution was set to 32 microsteps per step.

In order to get a reasonable injected volume on Tube, it was produced 10, 5, 4, 3 and 2 revolutions to stepper motor for 1 mL, 5 mL, 10 mL, 20 mL and 50 mL syringe, respectively. After obtaining the average volume injected, the revolution per mL factor was calculated for each syringe, and added to the Arduino sketch. Then, these train factors were tested by requesting a small volume injection. The volumes were measured and the average and percentage errors were analysed. The volume requested was 10% of the syringe maximum capabilities, except for 20 and 50 mL syringe, since this percentage would correspond to a volume higher than the Tube's recommended maximum usable volume (1.5 mL). In these two last cases, the volume injected was 1.5 mL. The results are presented in the following tables. Intermediate calculations may be found at Appendix B.

Table 18 - Revolutions per millilitre factor estimation results.

Syringe	Brand	rev/mL factor	Injected test volume (mL)	Test assays volume average (mL)	Standard deviation (mL)
1ml	BD	72.01053	0.100	0.10031	0.00055
5ml	Terumo	9.52014	0.500	0.50267	0.00319
10ml	HSW	6.34857	1.000	0.99889	0.00241
20ml	HSW	3.98688	1.500	1.49511	0.00916
50ml	Terumo	1.90529	1.500	1.50149	0.00514

The test assays were performed to verify if the revolution per millilitre factors was correctly estimated. The average volume injected was near the solicited volume and the standard deviation are satisfactory since for any syringe size it was obtained values less than 1% of variation. Furthermore, this result improves the ones obtained by MOST research group since they obtained 3% or less coefficient of variation for their NEMA17 syringe prototype. One of the factors that may be allowing this result improvement can be the 32 microsteps per step resolution used (they used a maximum of 16).

Table 19 presents the volume resolution that can be achieved depending on the syringe used. For all sizes, a resolution around or less than 1 µL per stepper motor step was obtained which seems a reliable value for a syringe pump. However, this resolution can be greatly increased when operating with higher microstepping resolutions.

Table 19 - Maximum volume resolution achieved for each syringe size.

Syringe size	Average maximum resolution (µL/step)	Average maximum resolution (nL/microstep)
1 mL	0.03472	1,09
5 mL	0.26260	8,21
10 mL	0.39379	12,31
20 mL	0.62706	19,60
50 mL	1.31214	41,00

Table 20 - Data to measure linear regression between Syringe ID^{-2} and rev/mL factor

Syringe	Syringe ID (mm)	rev/mL factor
1 mL	4.70	72.01053
5 mL	13.03	9.52014
10 mL	15.70	6.34857
20 mL	20.08	3.98688
50 mL	29.08	1.90529

It was obtained a linear regression with a reliable coefficient of determination (R^2) (Figure 48). The slope value of this linear regression was inserted in the source code in order to automatically convert the inner diameter inserted by the user into a consistent revolution per millilitre factor.

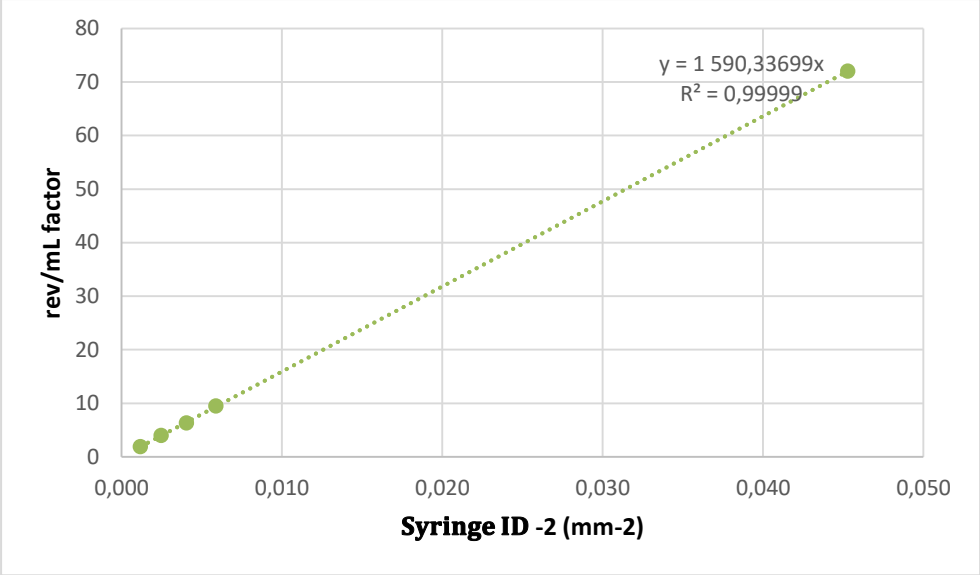
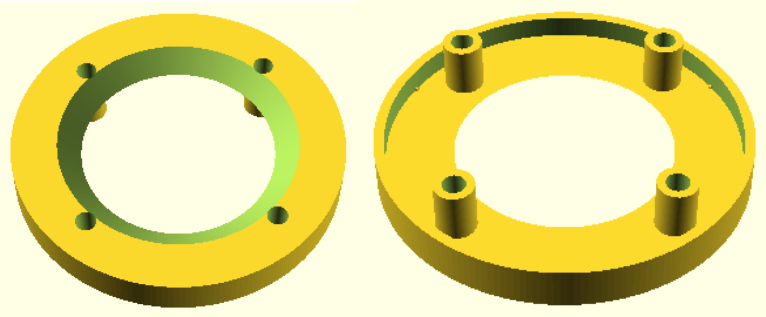


Figure 48 - Linear regression between Syringe ID^{-2} and rev/mL factor

4.3. 35 mm µ-Dish 3D printed Lid.

The µ-Dish used to perform the measures was the IBIDI 35 mm µ-Dish with high lateral walls.

Initially its original lid design was mimicked on the OPENSCAD software and then the modifications were performed. For tube's insertion points, a hole was created near the outside border of the lid, one for each tube. Then, a guide tube was designed to allow a better assembling between the fluid channels and the lid. To avoid obstructing the inner cell growth area, the lid has an open area to allow image acquisition. However, to maintain the culture chamber as sealed as possible a coverslip should fill this area gap. A small slope at the inner diameter edge was designed so the cover slip can be assembled and trapped in this inner zone. Figure 49 represents the final 3D-model aspect.



(a) (b)

Figure 49 - 35 mm µ-Dish Lid 3D model (a) top surface (b) bottom surface

The 3D printed lid was defined to have 4 holes for fluid exchange and the inner slope diameter was decided so it can hold a 25 mm diameter coverslip.

(a) (b) (c)

Figure 50 - 3D printed lid (a) bottom view (b) top view (c) lateral view

(a) (b) (c)

Figure 51 - Culture chamber assembling process: (a) original IBIDI 35 mm µ-Dish with original lid; (b) 25 mm diameter coverslip and 3D printed lid assembled to µ-Dish; (c) tube and coverslip assembling

4.4. Hardware Configuration procedure.

The final Arduino firmware is available at Appendix D.

After adding the C++ Arduino motor class to the Arduino device driver, the DLL file was compiled. The original mmgr_dal_Arduino.dll was replaced by this new one at main installing root.

The first task performed consisted on configuring the connected hardware components. The Hardware Configuration Wizard was opened and there, all the supported hardware drivers that are listed can be initialized if the physical component is already connected to computer. For the serial port communication devices, this connection must be performed before booting the software, otherwise the devices will not be recognized.

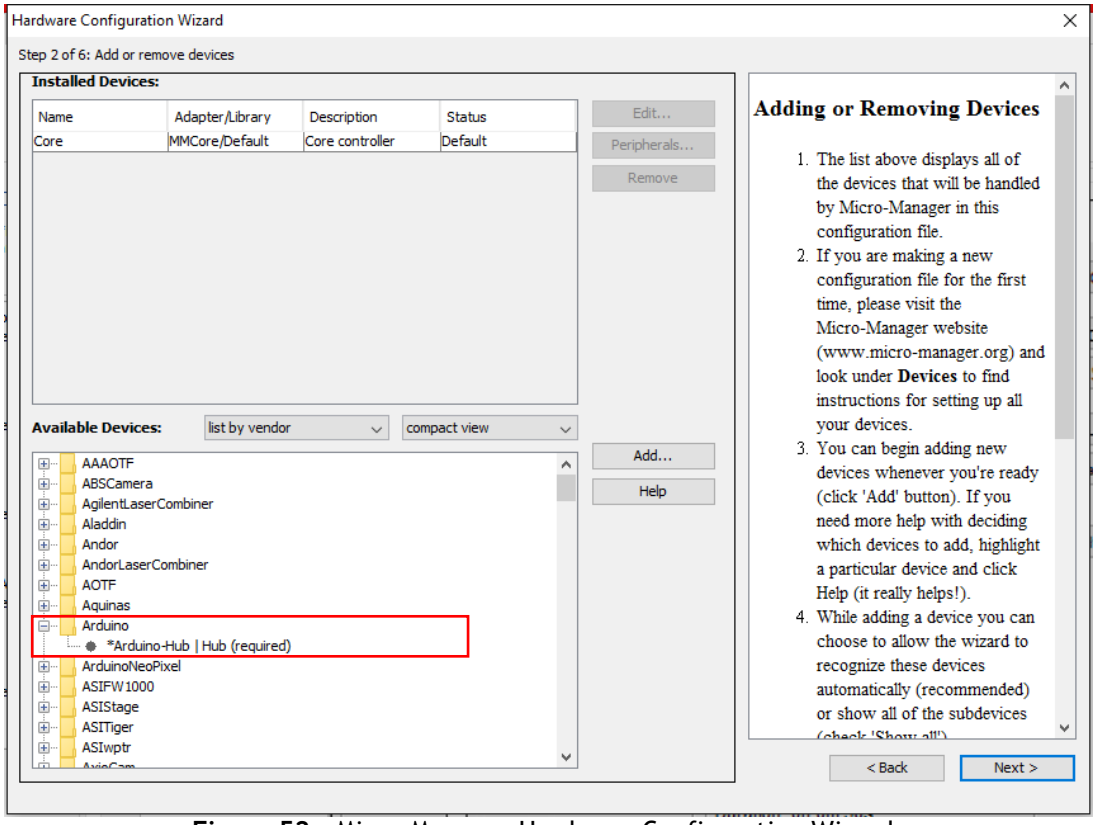


Figure 52 - Micro-Manager Hardware Configuration Wizard.

In this list, the Arduino device driver can be found. When selected, a new window will appear where some communication parameters can be defined, although these parameters are already predefined to work perfectly.

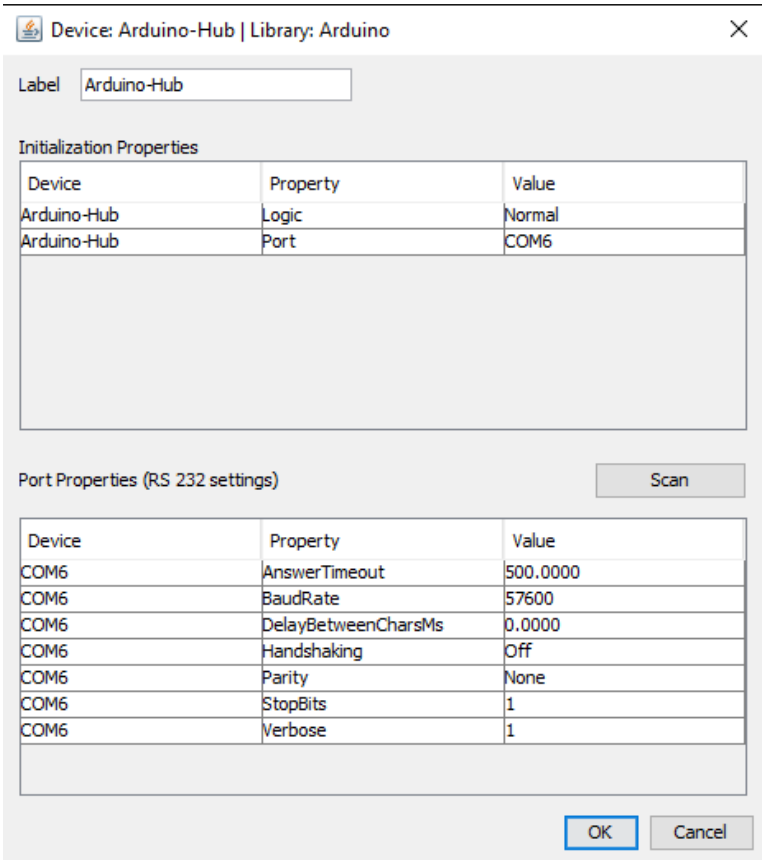


Figure 53 - Micro-manager and Arduino communication parameters.

After the communication parameters are validated, a list of the child devices that may be initialized appears. In this case, three new Arduino HUB child devices will appear: “Arduino-Motor1”, “Arduino-Motor2” and “Arduino-Motor3” (Figure 54).

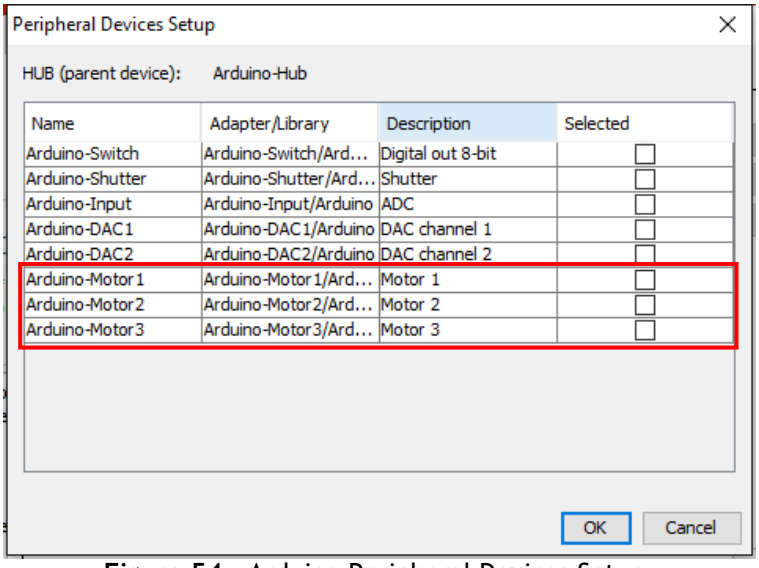


Figure 54 - Arduino Peripheral Devices Setup.

When an Arduino-Motor child device is selected, a new window opens, showing the user some parameters related with the syringe pump configuration adopted that should be pre-initialized, namely, information regarding the syringe that it is been used on the selected pump (Figure 55).

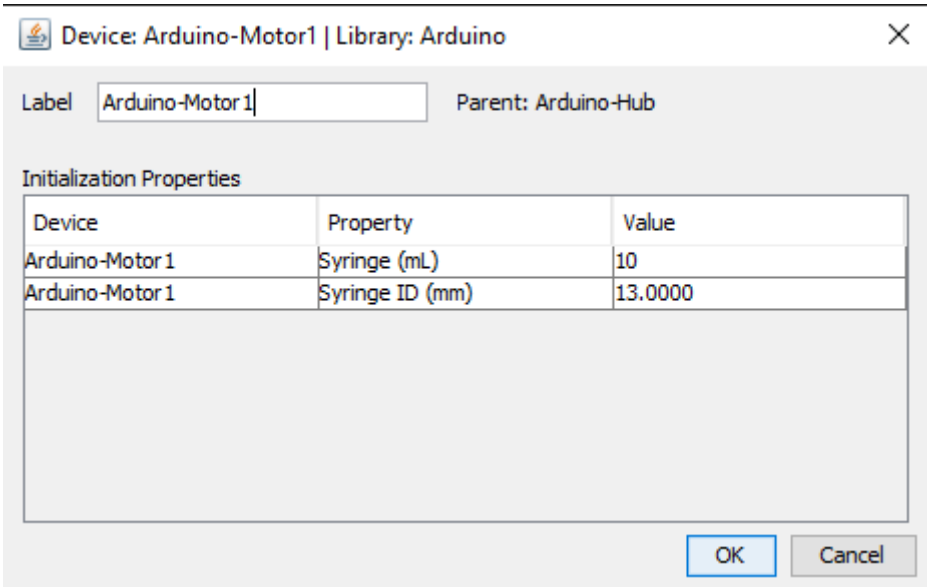


Figure 55 - Arduino Motor pre-initialized parameters.

This is a fundamental step since this property will define the revolution per millilitre factor that must be used to convert the future volume to be injected into an accurate stepper motor number of steps.

Finally, the Arduino HUB device and the child devices that were installed will appear at the installed devices list. Figure 56 demonstrates an example where all the three motors have been installed. The hardware configuration can be saved into a CFG extension file, in order to load it automatically when booting Micro-Manager.

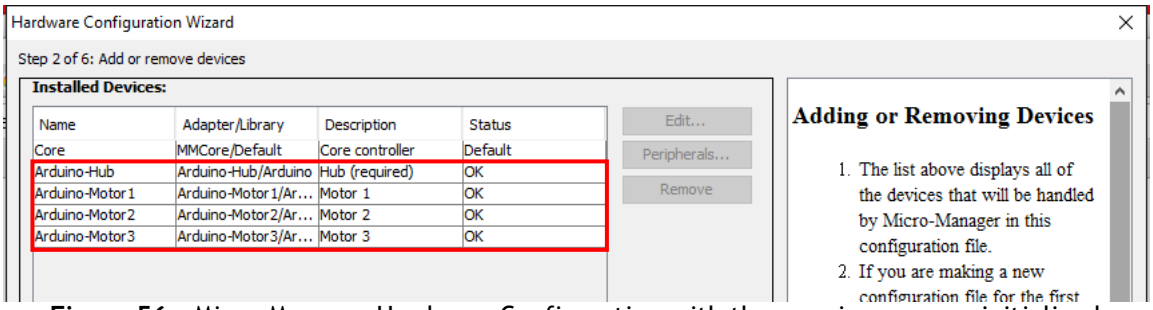


Figure 56 - Micro-Manager Hardware Configuration with three syringe pumps initialized.

Further, all the properties associated can be seen or changed at the Device Property Browser (Figure 57).

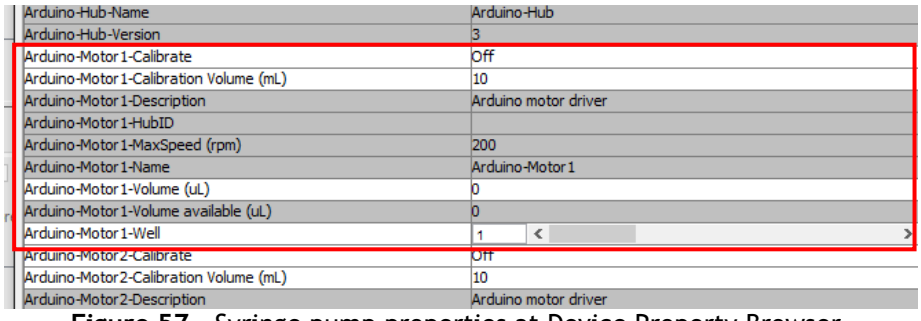


Figure 57 - Syringe pump properties at Device Property Browser.

4.5. Script environment

After being correctly installed, a syringe pump can be easily controlled by the Micro-manager Script Interface, without further source code modifications. There are already available some image acquisition script files to be downloaded at the Micro-manager official website. Using the properties defined section 3.1.2.2., it is possible to change the editable ones, namely the “Calibrate” and “Volume (uL)” properties, in order to achieve the pump task desired. This is possible since the mentioned properties trigger a syringe pump action. For this, the setProperty method can be used to change one of the referred properties. It accepts three input objects, with the following sequence: the device label, the property label and the property value that must be set.

For the calibration procedure, the property label is “Calibrate”, and this property allows only two values: “On” to start calibration process and “No” to ensure that this process is not initialized. This two allowed values have to be written in quotes since they were defined as a string object.

For a fluid exchange procedure, the property label is “Volume (uL)”, and this property must be set to the desired positive or negative amount of volume, in microliters, that should be infused or withdrawn, respectively.

As example, Figure 58 represents an illustrative syringe pump procedure, where the “Arduino-Motor1” syringe pump is initially calibrated. Posteriorly, 100 µL are infused and then 200 µL are withdraw.

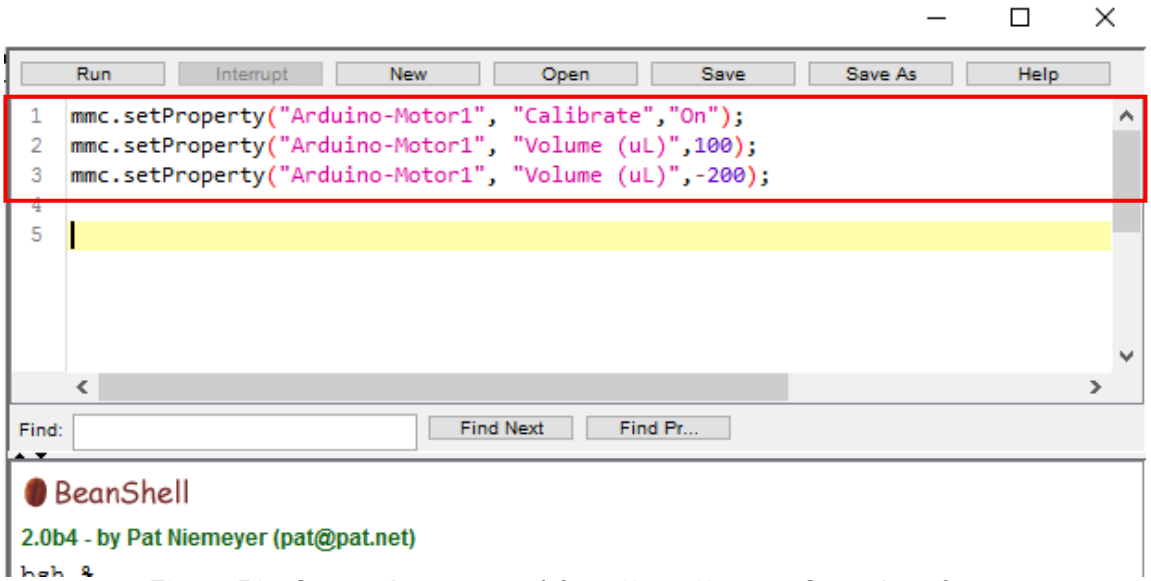


Figure 58 - Syringe Pump control from Micro-Manager Script Interface.

4.6. Multi-Dimensional Acquisition

To integrate the pump task capability tool to the Multi-Dimensional Acquisition engine, the java layer were initially edited and improved. The source code was built and debugged using NetBeans. The dialog was edited, adding a new checkbox panel to select whether it is wanted to perform or not a sequence of pump tasks during the acquisition protocol that is being set. The final aspect is represented in Figure 59.

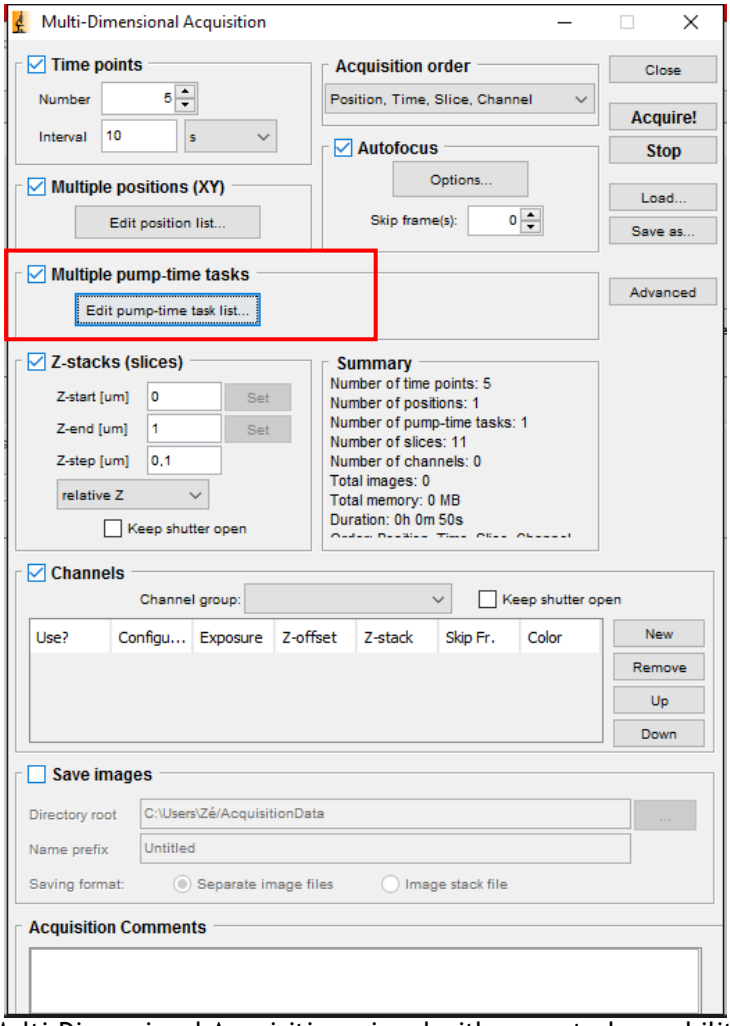


Figure 59 - Multi-Dimensional Acquisition wizard with pump task capabilities included.

As it can be seen, when selecting the checkbox, the “Edit pump-time task list…” button inside this pane becomes clickable and redirects the user to create or edit the pump task sequence that should be performed. A new window opens and this dialog was created having into account the requirements stated in section 3.1.2.3. A table appears on the left side and some auxiliary components are represented on the right side.

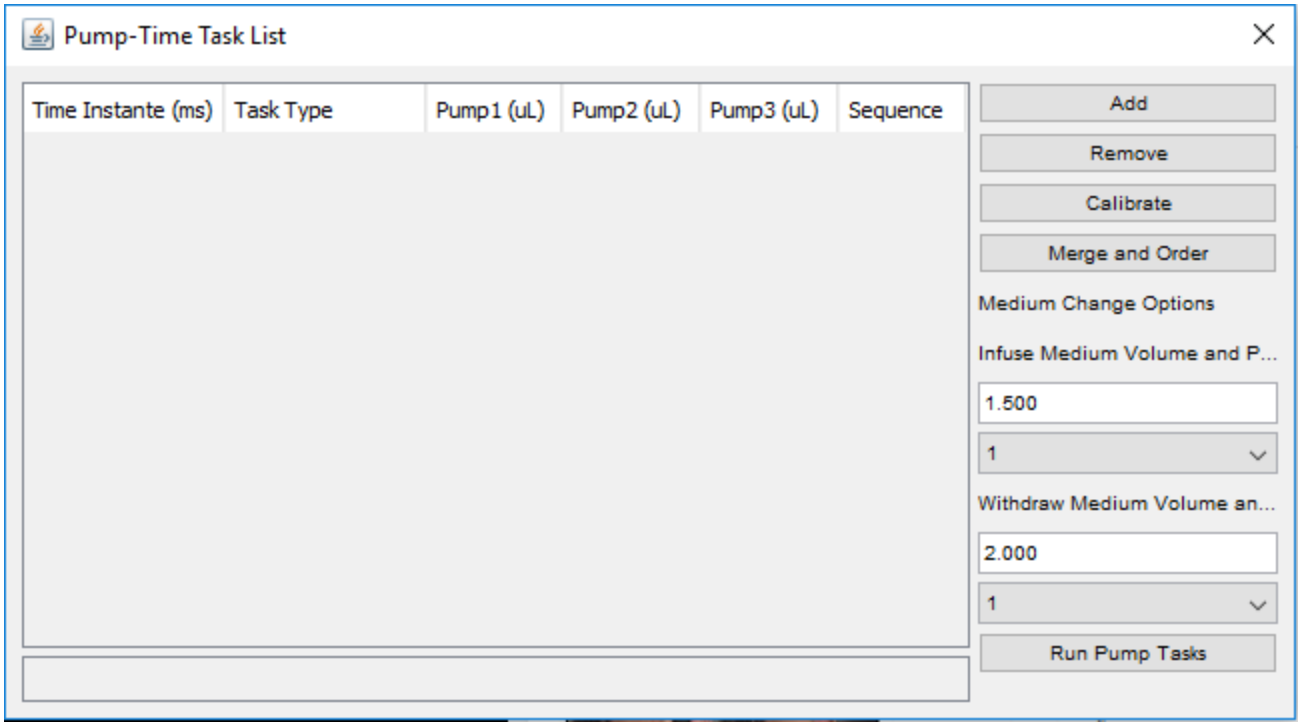


Figure 60 - GUI created to define the pump task sequence desired.

When clicking the "Add" button, a new row will be added to the table as exampled in Figure 61. This row represents a pump task and its cells may be editable except the "Pump1", "Pump2" and "Pump3" which depends if the correspondent Arduino child device has already been correctly configured. "Pump1", "Pump2" and "Pump3" represent "Arduino-Motor1", "Arduino-Motor2" and "Arduino-Motor3" child device labels, respectively.

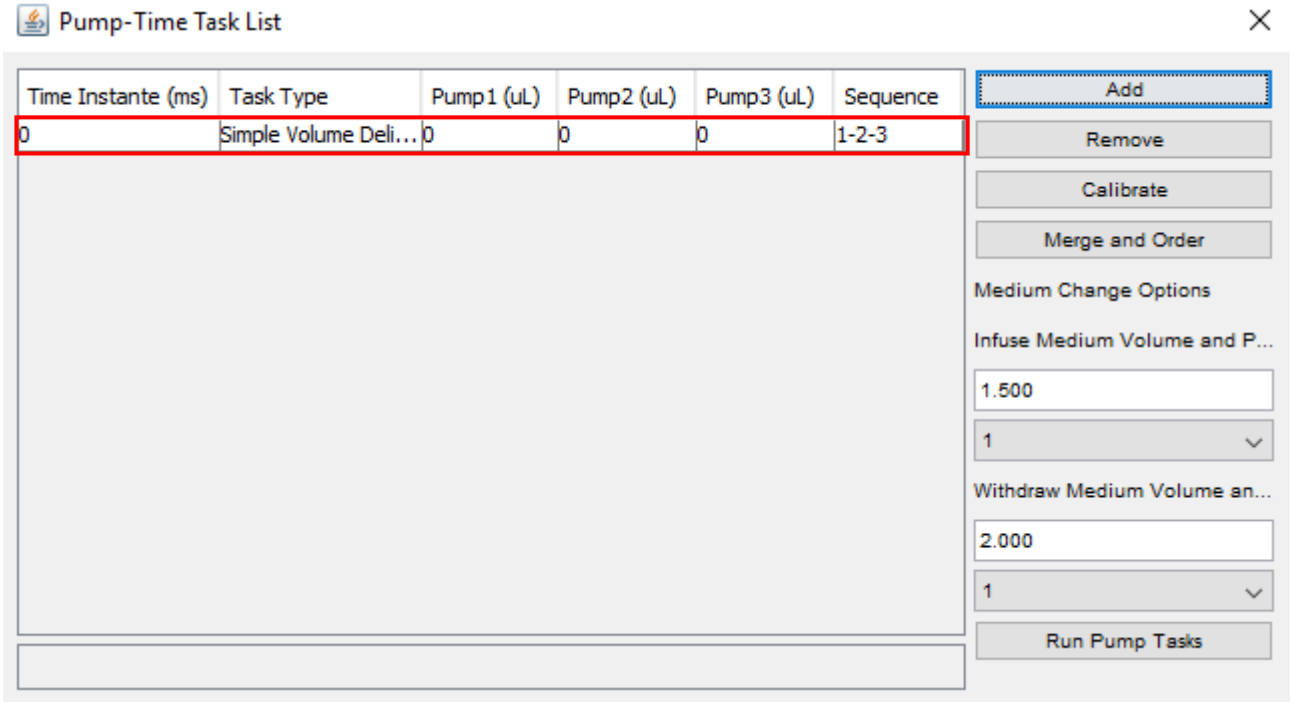


Figure 61 - New pump task instance creation.

The "Remove" button removes the current table row selected.

A "Calibrate" button was created to trigger the calibration of the pumps already configured. When selected, the user is asked to confirm its decision to calibrate the syringe pumps. Then, a dialog opens so the user can insert the volume, which the syringe pump will be calibrated. This is an important step since this value

will help to manage the software information about the current volume contained inside the syringe during the pump task sequence. Hence, the software has the ability to abort the pump task sequence if making a certain pump task would bring damage to the syringe pump mechanism.

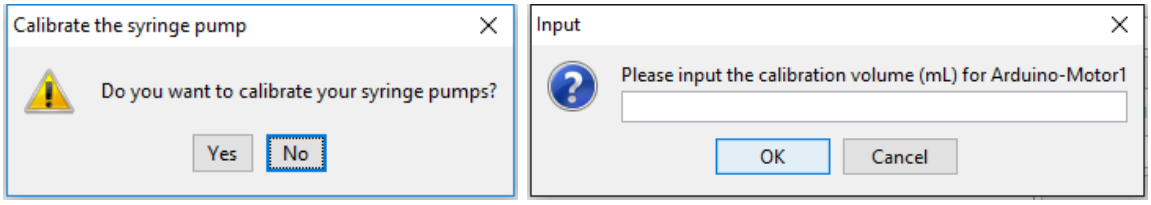
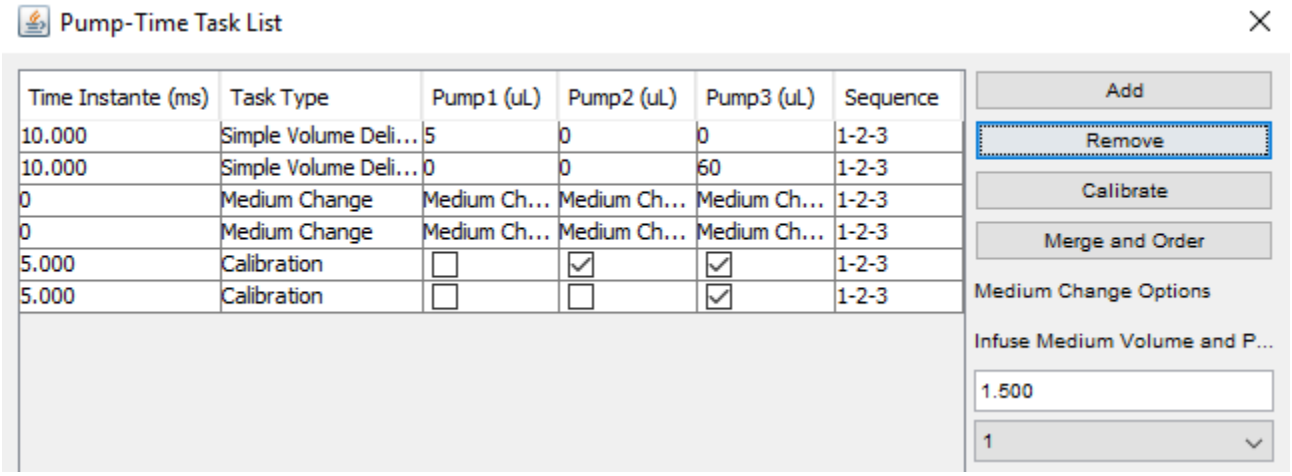
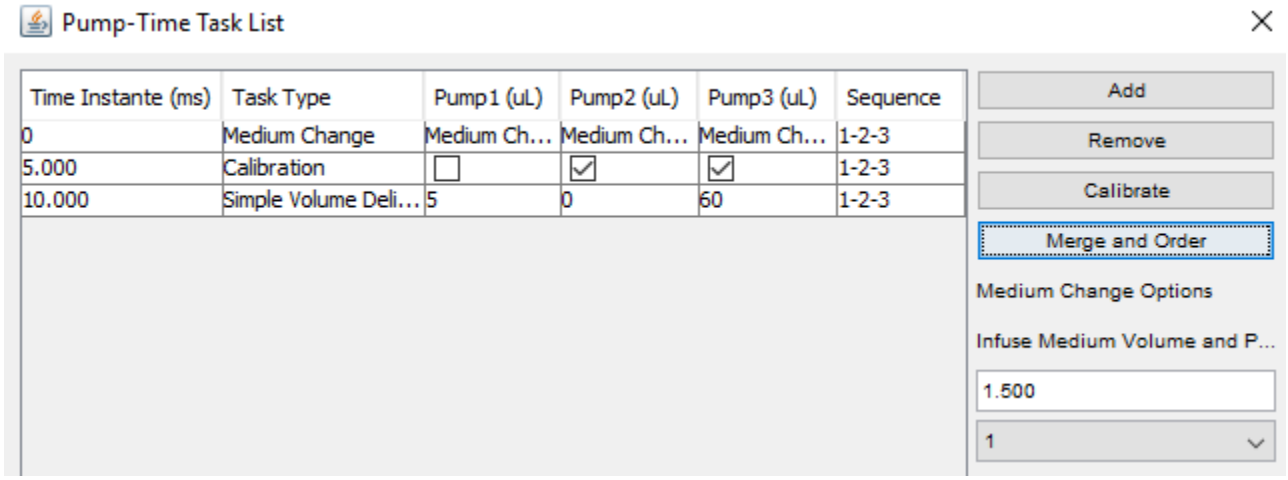


Figure 62 - Calibration procedures dialogs.

The merge and order button organizes the pump task table in order to set a more useful of the pump task sequence to the user. In this process, pump tasks of the same type and to be performed at a same time are merged and the information contained at “Pump1”, “Pump2” or “Pump3” is combined. Then, the pump task list is ordered by time. An example is shown below.



(a)



(b)

Figure 63 - "Merge and Order" button example procedure: (a) before (b) after

In the bottom on the right side of the windows, the medium change pump task type can be configured. It can be defined how much volume should be exchanged and which pump should perform each medium change task in a combo box.

Finally, a pump task sequence can run outside the multi-dimensional image acquisition engine. This can be done when the experimental protocol that is being followed does not require image acquisition during the pump task sequence or for debug purpose. This window can be also opened on “Tools”->”Pump time task list…” at main Micro-manager software window.

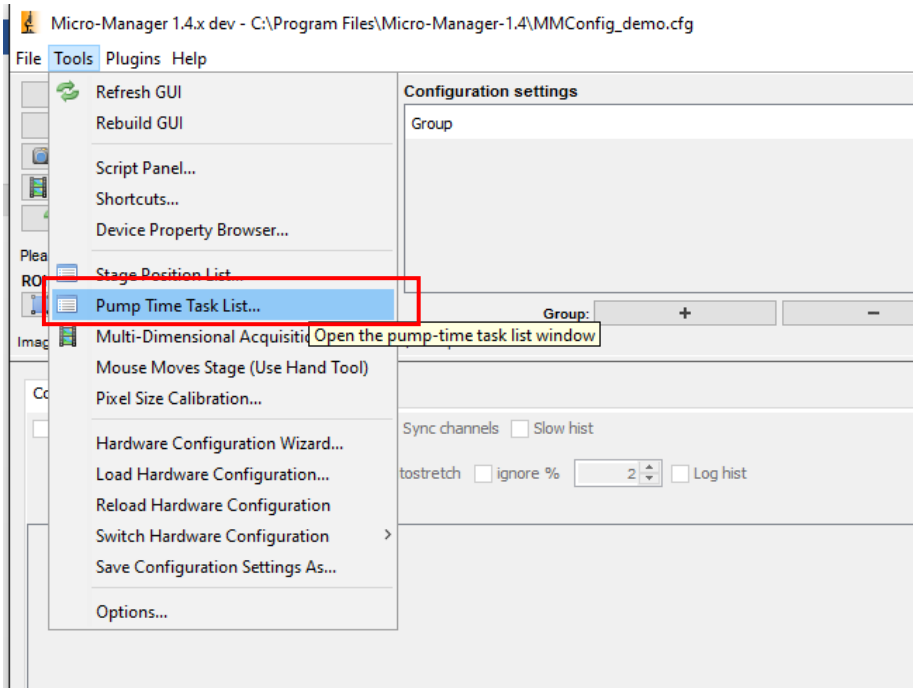


Figure 64 - Open pump-time task list from main Micro-Manager GUI.

4.7. Proof of Concept

As previously mentioned in Chapter 3, the designed solution should be validated, namely for live cell imaging applications. Two experiments were performed: one to test the culture medium replacement procedure and another for simple volume delivery at a specific time instant.

During the first experiment, 6 medium replacement procedures were initially performed, equality spaced by 2 minutes. Then, it was defined that the time-lapse image should record the cell activity during 12 hours, with 5 minutes interval between frames. Furthermore, the volume infused was defined as 2000 µL and the volume withdraw as 3500 µL to ensure that all the culture medium contained at the culture chamber was removed. Thus, 20 mL and 50 mL syringes were attached to infuse and withdraw syringe pumps respectively.

In the second experiment, a simple volume delivery task was performed. Since the volume to be delivered was 200 µL, a 1 mL syringe was chosen. The results of both experiments are described below.

For both protocols, the designed solution was set near the Zeiss Axiovert microscopy (Figure 65). The syringe pumps were set outside the microscope environmental chamber and silicon tubes guided the fluids between the syringes and the IBIDI µ-Dish. The IBIDI µ-Dish was also sealed inside the stage chamber in order to maintain cell culture at 37 ºC and at 5% CO_2 (detail at Figure 65).

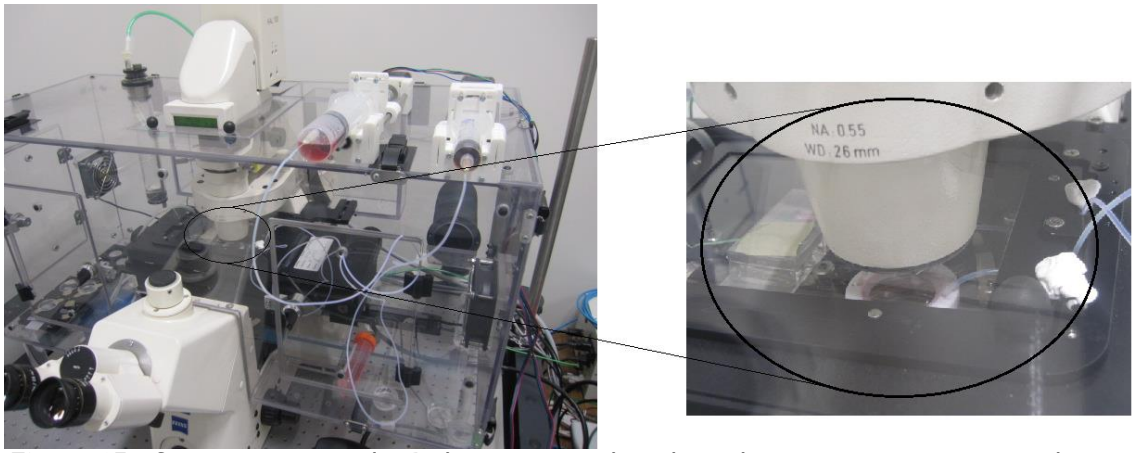


Figure 65 - Syringe pumps and µ-Dish position in the adopted microscopy environment layout

The pump task table inserted at the µ-Manager MDA tool for both experiments are shown in Figure 66 and Figure 67, respectively.

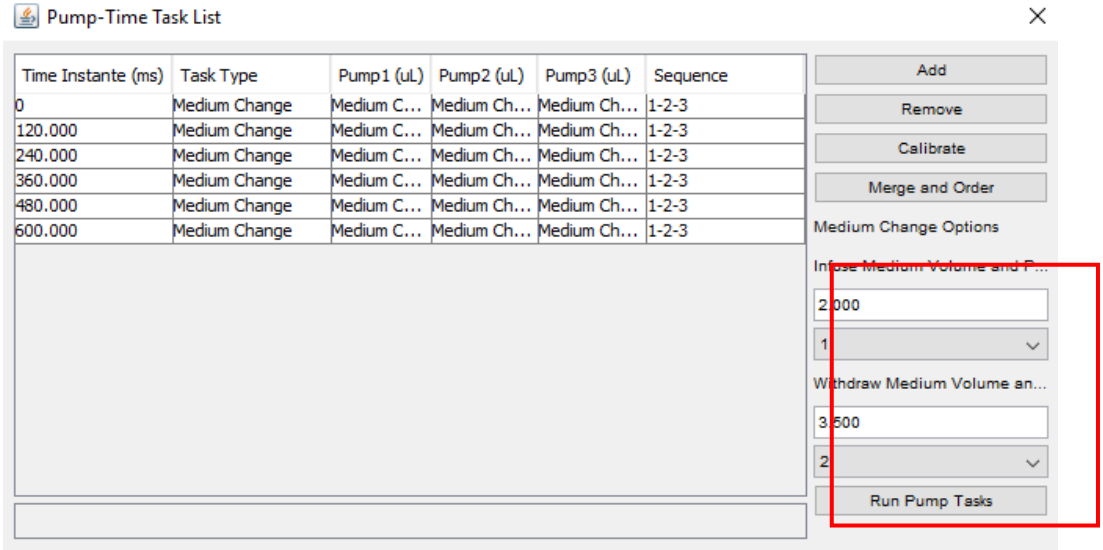


Figure 66 - Pump task table defined for the first proof of concept protocol

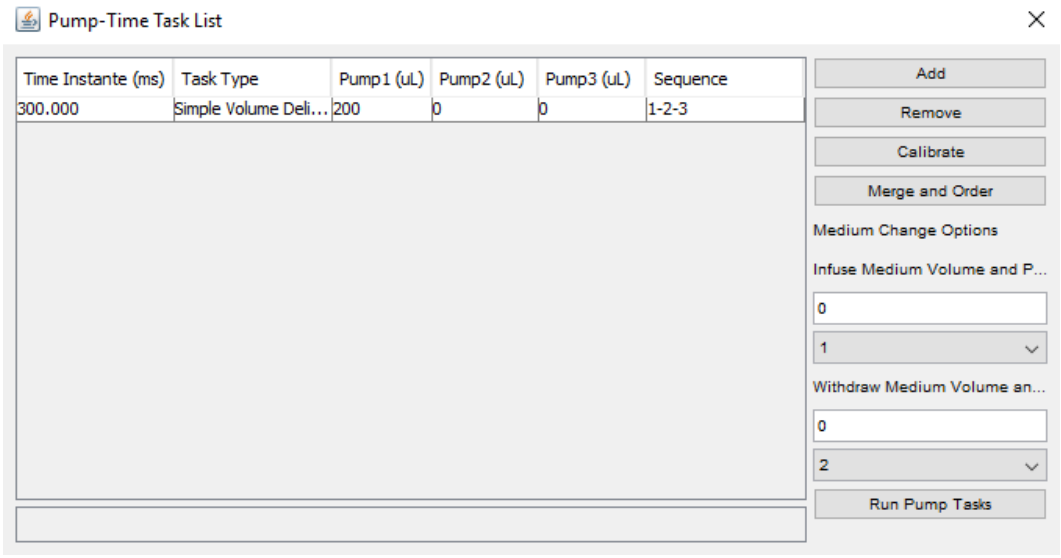


Figure 67 - Pump task table defined for the second proof of concept protocol

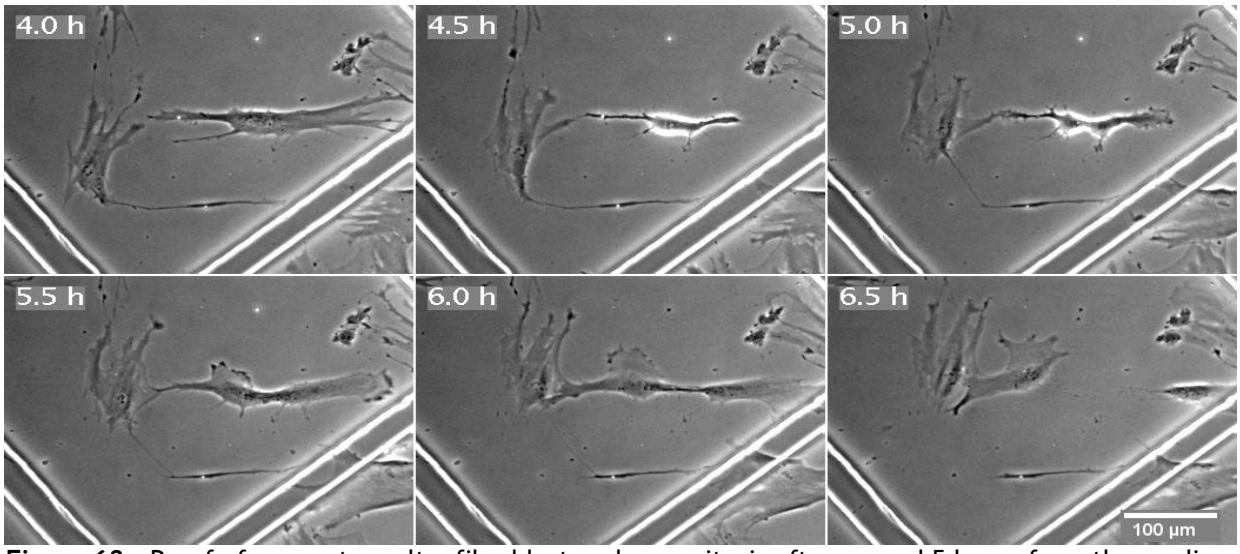


Figure 68 - Proof of concept results: fibroblast undergo mitosis after around 5 hours from the medium replacement procedures (Timestamp: 4,0h; 4,5h; 5,0h; 5,5h; 6,0h and 6,5h after pump tasks)

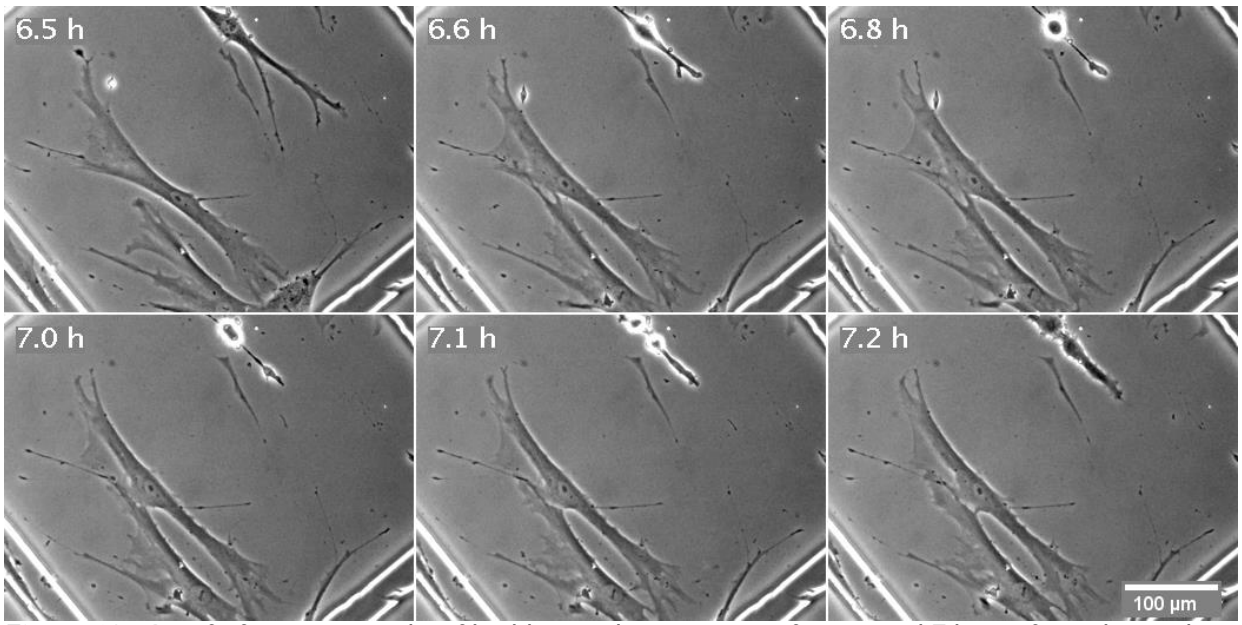


Figure 69 - Proof of concept results: fibroblast undergo mitosis after around 7 hours from the medium replacement procedures (Timestamp: 6,5h; 6,6h; 6,8h; 7,0h; 7,1h and 7,2h after pump tasks)

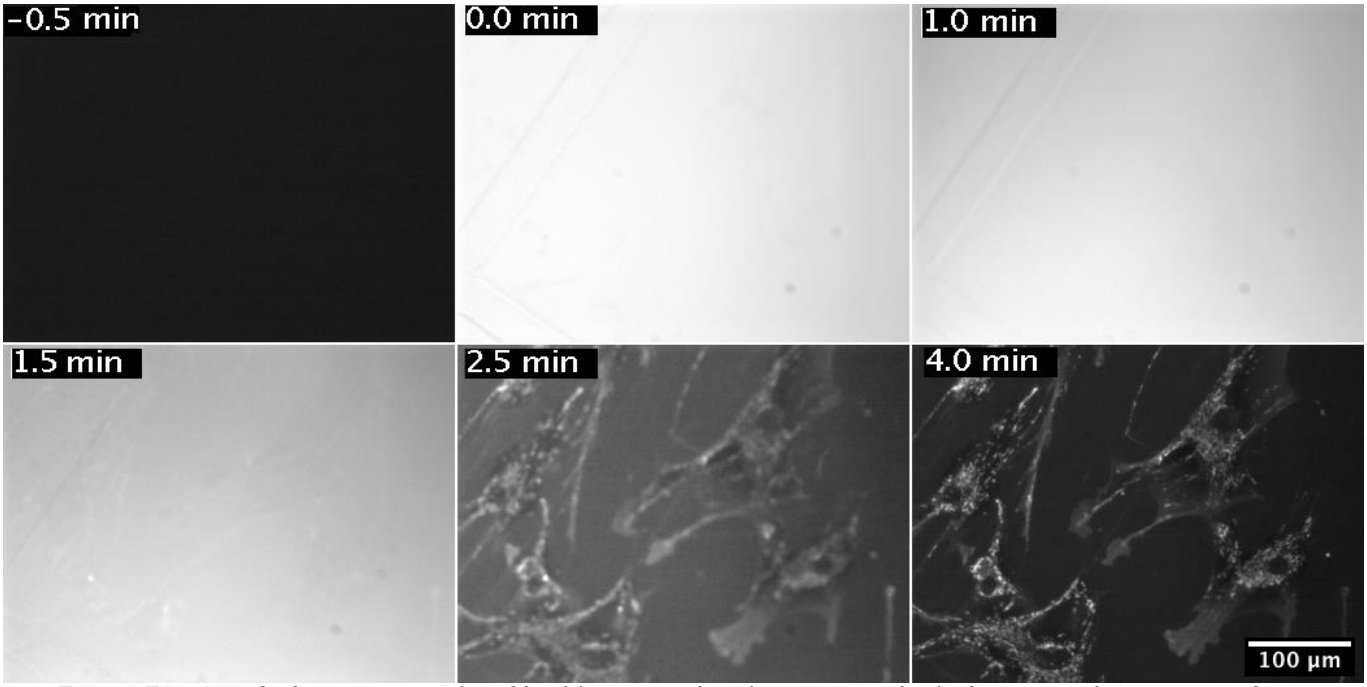


Figure 70 - Proof of concept results: fibroblast mitochondrias are marked after around 4 minutes after fluorescence marker infusing (Timestamp: -0,5min; 0,0min; 1,0min; 1,5min; 2,5min and 4,0min after infusion)

As it can be seen in Figure 68 and Figure 69, the cells resumed their normal mitosis cycle and duplicated themselves around 5 and 7 hours after performing the 6 fresh medium replacements, respectively. These results support the viability of the designed system since similar results can be obtained using the syringe pump solution, when compared with the manual medium replacement procedure.

Moreover, it is possible to observe mitochondrias in fibroblast around 2 minutes after fluorescence marker infusion. The figures presented in Figure 70 are displayed with the same linear histogram stretching. After infusing the marker into the culture chamber, the image grey levels highly increase, as denoted by the full white image shown in Figure 70 b). Posteriorly, the grey levels gradually decrease along time, since the marker is spreading all over the culture chamber area. Finally, the marker penetrates the cell's cytoplasmic membrane, and the mitochondria positions inside the cells can be detected.

Chapter 5

Conclusions and Future Work

Nowadays, cell's behaviour is widely studied, namely the changes that occur when they face different kind of stimuli. Normally, to capture this kind of phenomena live cell imaging and fluid delivery systems are often used. There are already fluid delivery systems available on market that allows a controlled and automated fluid exchange with culture chambers. However, they represent an expensive investment for research institutions.

A lot of open-source technologies are already being developed and Micro-Manager is a good example, since it integrates several kinds of microscopy hardware from various companies. Furthermore, open-source platforms can greatly decrease technology costs.

Thus, this project aimed to enhance the Micro-manager software allowing it to control a range of open-source syringe pumps, in order to reach a controlled fluid infusing and withdrawing on a culture chamber. Not only MOST syringe pumps were adapted to fulfil the desired application, but a proper lid for 35 diameter µ-dish culture chamber was also designed to house the silicone tubes that guide the fluids.

Lastly, the final solution was tested with two proof of concept experiments. The first protocol tested if the medium replacement procedure was properly implemented, and the second one verified the infusing of an amount of volume at a specific time instant.

5.1. Achievements

A MOST syringe pump was adapted to fulfil the project requirements, namely for being used during live cell imaging experiments. Furthermore, a lid for 35 diameter µ-dishes was specifically designed, in order to house the silicon tubes that guide the fluids from the syringes to the culture chamber, and vice versa. A total of three syringe pumps were assembled.

An electronic circuit was designed to control the syringe pumps. This circuit was controlled by an Arduino board, which intermediates the communication between the Micro-Manager software and the syringe pumps.

Syringes with the same maximum volume capability can vary in dimensions. To avoid measuring a revolution per millilitre factor for each unique syringe, a linear regression that relates the syringe inner diameter with a revolution per millilitre factor was measured. This allowed the final user to use syringe ranging from 1 to 50 mL volume capability from several brands.

The Micro-manager software had already the ability to communicate with Arduino boards. However, the device driver and the Arduino firmware were not designed to control stepper motors, which drive the syringe pumps. Thus, a range of new possible commands was defined in both source codes mentioned before.

A new GUI was implemented in order to allow the user easily insert the desired syringe pump tasks. The pump tasks are inserted and displayed in a list form, and calibration procedures must be performed before running a pump tasks sequence.

This new Micro-manager GUI was also connected to the MDA tool, in order to run live cell imaging protocols with pump tasks at specific time instances.

Two proof of concepts protocols tested the final solution, which confirmed reliable results and a very good performance, as expected.

5.2. Future work

This work represents the first steps of an ambitious project. A lot of improvements and new functionality can be developed in order to achieve a better and suitable solution. Some of these improvements were identified during the project development.

The syringe pump was tested in order to measure its maximum resolution achievable since it was expected an accurate volume dispensing for such small volume. However, further tests should be performed so the syringe pump may be better described. It should be measured maximum speed for each syringe and studied how fluid viscosity can affect syringe pump operation.

The designed electronic circuit was prototyped in a breadboard. Due to the high number of electronic hardware parts that were assembled, namely the wires, sometimes some of them stop connecting the correct place and a frequent check had to be executed in order to correct these mistakes. Further, stepper motors are very sensitive to involuntary wire disconnections, which can lead to irreparable damages to these motors. Thus, it would be valuable to design a PCB to better secure these hardware components. In fact, designing it as an Arduino shield can also be an excellent option.

Furthermore, a touchscreen may be used to replace the LCD and push-buttons.

During the proof of concept experiments, it was verified that the culture chamber sealing with the designed lid mechanism (lid plus coverslip) was not the best solution for experiments that require long time image acquisition. The mechanism could not seal the culture chamber in a proper way since the medium evaporated after around 10 hours. This phenomenon was also promoted by the 37º C environment. Further, other sealing possibilities should be considered and tested, in order to avoid this problem. Since the fluid delivery system is already prototyped, it would be interesting to repeat the proof of concept experiments using a perfusion cell culture. This cell cultures have a better sealing mechanism that can lead to less contamination events at long term experiments.

Lastly, the solution achieved let the syringe pump to guide fluids only to one final acceptor. Some experiments are performed in multi-well plates, which require delivering fluids to more than one culture chamber. So, the designed solution is not suitable for this kind of experiments since it would be necessary to build a higher number of syringe pumps. This problem can be overcome with an automated valve system between the syringe pump and the culture chamber. This system should be designed to accept one single input to connect to syringe and multiple outputs to connect each well of the multi-well plate. Furthermore, the valve system should also be controlled by Micro-manager software.

References

[1] "Micro-Manager." [Online]. Available: https://www.micro-manager.org/. [Accessed: 05-Feb-2016].

[2] M. Abramowitz, "Microscopy: Basics and Beyond," *Mol. Expressions Microsc. Prim.*, vol. 1, pp. 1–50, 2003.

[3] Olympus, *Basics of Light Microscopy & Imaging*. .

[4] Y. Hamaguchi, "Optical microscopy," *Tanpakushitsu Kakusan Koso.*, vol. 42, no. 7 Suppl, pp. 1026–1032, 1997.

[5] "What is Light Microscopy?" [Online]. Available: https://www.jic.ac.uk/microscopy/intro_LM.html. [Accessed: 05-Feb-2016].

[6] M. M. Frigault, J. Lacoste, J. L. Swift, and C. M. Brown, "Live-cell microscopy - tips and tools," *J. Cell Sci.*, vol. 122, no. 6, pp. 753–767, 2009.

[7] "Perfusion for live cell imaging: Methods and techniques - Elveflow." [Online]. Available: http://www.elveflow.com/microfluidic-tutorials/cell-biology-imaging-reviews-and-tutorials/live-cell-perfusion/methods-and-techniques/. [Accessed: 21-Jan-2016].

[8] Z. Jiang and T. Veitinger, "Live-cell Imaging Techniques," 24-Feb-2012. [Online]. Available: http://www.leica-microsystems.com/science-lab/live-cell-imaging-techniques/. [Accessed: 06-Feb-2016].

[9] "Cell Culture Environment on the Microscope :: ibidi." [Online]. Available: http://ibidi.com/applications/live-cell-imaging/cell-culture-environment-on-the-microscope/#c562. [Accessed: 21-Jan-2016].

[10] "HOME Microfluidic flow control - Elveflow." [Online]. Available: http://www.elveflow.com/. [Accessed: 27-Jan-2016].

[11] "OB1 - 4 channels microfluidic flow controller - Elveflow." [Online]. Available: http://www.elveflow.com/microfluidic-flow-control-products/flow-control-system/pressure-controller/. [Accessed: 27-Jan-2016].

[12] "MUX - microfluidic flow switch matrices - Elveflow." [Online]. Available: http://www.elveflow.com/microfluidic-flow-control-products/flow-control-system/flow-multiplexer/. [Accessed: 10-Feb-2016].

[13] "Company Overview :: ibidi." [Online]. Available: http://ibidi.com/about-ibidi/profile/company-overview/. [Accessed: 27-Jan-2016].

[14] "ibidi Pump System." [Online]. Available: http://ibidi.com/xtproducts/en/Instruments-Accessories/ibidi-Pump-System. [Accessed: 27-Jan-2016].

[15] "CellASIC® ONIX Live Cell Analysis | Life Science Research | Merck Millipore." [Online]. Available: http://www.merckmillipore.com/PT/en/life-science-research/cell-culture-systems/cellASIC-live-cell-analysis/d1Cb.qB.w58AAAE_0j13.MnA,nav. [Accessed: 28-Jan-2016].

[16] H. Gate, S. G. Uk, J. Williams, and D. Hill, *World Precision Instruments 1*, no. 4015. 2014.

[17] "Standard Infuse/Withdraw PHD Ultra Syringe Pumps." [Online]. Available: https://www.harvardapparatus.com/webapp/wcs/stores/servlet/product_11051_10001_62851_-1_HAI_ProductDetail___#fulldescriptiontab. [Accessed: 05-Feb-2016].

[18] HNP Mikrosysteme, "Operation Principle," 03-Feb-2016. [Online]. Available: http://www.hnp-mikrosysteme.de/en/products/technical-information/operation-principle.html. [Accessed: 05-Feb-2016].

[19] HNP Mikrosysteme, "Micro annular gear pump mzr -4605," no. 02, pp. 6292–6293.

[20] A. B. Park and O. Road, "Electro-osmotic Pump Cores and Accessories," pp. 1–8.

[21] A. B. Park and O. Road, "Piezoelectric Pumps," pp. 1–9.

[22] “CurieJet® Liquid Micro Pump :: Micro Fluidics Pump, Micropump.” [Online]. Available: http://www.curiejet.com/en/products/list.php?pin=099861a6e97183485912f85df4e401fd&type=s. [Accessed: 05-Feb-2016].
[23] “Piezoelectric Micropumps for Pumping and Metering Technology.” [Online]. Available: http://www.piceramic.com/applications/piezo-micropumps.html. [Accessed: 05-Feb-2016].
[24] “Open Hardware Microfluidics Controller Arduino Shield — StreyLab.” [Online]. Available: http://streylab.com/blog/2015/4/8/open-hardware-microfluidics-controller-arduino-shield. [Accessed: 04-Feb-2016].
[25] “Ardulink,” *What is*. [Online]. Available: http://www.ardulink.org/what-is/.
[26] Ardulink, “Ardulink,” *Automatic Lipid Dispensing*, 2014. [Online]. Available: http://www.ardulink.org/automatic-lipid-dispensing/.
[27] S. Balakrishnan, M. S. Suma, S. R. Raju, S. D. B. Bhargav, S. Arunima, S. Das, and G. K. Ananthasuresh, “A Scalable Perfusion Culture System with Miniature Peristaltic Pumps for Live-Cell Imaging Assays with Provision for Microfabricated Scaffolds,” *Biores. Open Access*, vol. 4, no. 1, pp. 343–357, 2015.
[28] B. Wijnen, E. J. Hunt, G. C. Anzalone, and J. M. Pearce, “Open-source syringe pump library,” *PLoS One*, vol. 9, no. 9, pp. 1–8, 2014.
[29] Micro-Manager, “Micro-Manager Project Overview.” [Online]. Available: https://micro-manager.org/wiki/Micro-Manager Project Overview.
[30] “Instructions µ-Dish 35mm, high.”
[31] “Lynch open source syringe pump modifications,” 2015. [Online]. Available: http://www.appropedia.org/Lynch_open_source_syringe_pump_modifications.
[32] “42BYGHM809 Datasheet.” .
[33] Polulu Robotics & Eletronics, “DRV8834 Low-Voltage Stepper Motor Driver Carrier.” [Online]. Available: https://www.pololu.com/product/2134.
[34] Micro-Manager, “Building MM on Windows.” [Online]. Available: https://micro-manager.org/wiki/Building_MM_on_Windows#Before_you_start.
[35] “Arduino - Micro-Manager.” [Online]. Available: https://www.micro-manager.org/wiki/Arduino. [Accessed: 05-Feb-2016].
[36] “Building Micro-Manager Device Adapters.” [Online]. Available: https://micro-manager.org/wiki/Building_Micro-Manager_Device_Adapters.
[37] Cadence Science, “Custom Made Matched Point Needle Sets.” [Online]. Available: http://cadenceinc.com/catalog/product-group/custom-made-matched-point-sets/.

Appendix A

Literature Review comparison tables

Table A 1 - Comparison between perfusion systems

	Pressure induced perfusion	Peristaltic Pumps	Syringe Pumps
Advantages	Very stable Pulse-less flow rate Sterilisable Wide range of pressure operation	Pumps continuous volumes Sterilisable Less expensive for multiple channel dispensing 1 to 16 channels of operation Can dispense and withdraw	Works at higher pressures Higher precision Pulse free flow Accurately dispense very small to large volumes Easily sterilisable Can dispense or withdraw
Disadvantages	Slightly more expensive	Low pressure operation Pulsing flow Moderate precision	Pumps finite volumes Slightly more expensive
Main applications		Perfusion flow across tissue or cells Pump in and out with balanced flow Transfer bulk liquids e.g., controlled animal feeding	Pumping sample/calibrant into Mass Spectrometer Accurate dispensing of drugs in animals High pressure flow into reaction chamber

Appendix B

Revolution per millilitre factor results

Table B 1 - Revolution per millilitre factor results - 1 mL syringe

Train assays	Empty Tube	Tube + volume injected after 10 revolutions (g)	Volume of 10 revolutions (g)	Volume of 10 revolutions (mL)
1	0,95455	1,09272	0,13817	0,137869
2	0,94485	1,0845	0,13965	0,139346
3	0,94133	1,08077	0,13944	0,139136
4	0,93697	1,07591	0,13894	0,138637
5	0,95347	1,09313	0,13966	0,139356
			Average volume	0,138869
			Standard deviation	0,000493
			Factor	72,01053

Test assays	Empty Tube	Tube + volume injected of 0,1 mL (g)	Volume injected of 0,1 mL (g)	Volume injected of 0,1 mL (mL)
1	0,93687	1,03653	0,09966	0,099443
2	0,93879	1,0388	0,10001	0,099792
3	0,94521	1,04597	0,10076	0,10054
4	0,94495	1,0457	0,10075	0,10053
5	0,95451	1,05597	0,10146	0,101239
			Average volume	0,100309
			Standard deviation	0,000553
			Factor	71,78883

Table B 2 - Revolution per millilitre factor results - 5 mL syringe

Train assays	Empty Tube	Tube + volume injected after 5 revolutions (g)	Volume of 5 revolutions (g)	Volume of 5 revolutions (mL)
1	0,94734	1,4707	0,52336	0,522219
2	0,94497	1,46585	0,52088	0,519744
3	0,94736	1,4761	0,52874	0,527587
4	0,93885	1,46919	0,53034	0,529184
5	0,94527	1,4737	0,52843	0,527278
			Average volume	0,525202
			Standard deviation	0,003377
			Factor	9,520139
Test assays	Empty Tube	Tube + volume injected of 0,5 mL (g)	Volume injected of 0,5 mL (g)	Volume injected of 0,5 mL (mL)
1	0,94342	1,43935	0,49593	0,494849
2	0,95579	1,46695	0,51116	0,510046
3	0,94209	1,44587	0,50378	0,502682
4	0,94754	1,45191	0,50437	0,50327
5	0,94445	1,44807	0,50362	0,502522

Average volume	0,502674
Standard deviation	0,003191
Factor	9,469502

Table B 3 - Revolution per millilitre factor results - 10 mL syringe

Train assays	Empty Tube	Tube + volume injected after 4 revolutions (g)	Volume of 4 revolutions (g)	Volume of 4 revolutions (mL)
1	0,94486	1,5778	0,63294	0,63156
2	0,94695	1,58252	0,63557	0,634184
3	0,94444	1,5762	0,63176	0,630383
4	0,95587	1,58619	0,63032	0,628946
5	0,94513	1,57174	0,62661	0,625244
			Average volume	0,630063
			Standard deviation	0,002375
			Factor	6,348568
Test assays	Empty Tube	Tube + volume injected of 1 mL (g)	Volume injected of 1 mL (g)	Volume injected of 1 mL (mL)
1	0,95039	1,9518	1,00141	0,999227
2	0,9575	1,95942	1,00192	0,999736
3	0,93476	1,9298	0,99504	0,992871
4	0,94733	1,95087	1,00354	1,001352
5	0,93691	1,94035	1,00344	1,001252
			Average volume	0,998887
			Standard deviation	0,002407
			Factor	6,355639

Table B 4 - Revolution per millilitre factor results - 20 mL syringe

Train assays	Empty Tube	Tube + volume injected after 3 revolutions (g)	Volume of 3 revolutions (g)	Volume of 3 revolutions (mL)
1	0,942	1,70092	0,75892	0,757265
2	0,94661	1,71454	0,76793	0,766256
3	0,93461	1,68132	0,74671	0,745082
4	0,94212	1,69569	0,75357	0,751927
5	0,94671	1,69014	0,74343	0,741809
			Average volume	0,752468
			Standard deviation	0,007434
			Factor	3,986881
Test assays	Empty Tube	Tube + volume injected of 1,5 mL (g)	Volume injected of 1,5 mL (g)	Volume injected of 1,5 mL (mL)
1	0,94669	2,45662	1,50993	1,506638
2	0,95343	2,44301	1,48958	1,486332
3	0,93466	2,44444	1,50978	1,506488
4	0,95582	2,44471	1,48889	1,485644
5	0,94514	2,43883	1,49369	1,490433
			Average volume	1,495107
			Standard deviation	0,009165
			Factor	3,999929

Table B 5 - Revolution per millilitre factor results - 50 mL syringe

Train assays	Empty Tube	Tube + volume injected after 2 revolutions (g)	Volume of 2 revolutions (g)	Volume of 2 revolutions (mL)
1	0,94529	1,99708	1,05179	1,049497
2	0,94975	2,0047	1,05495	1,05265
3	0,94968	2,00417	1,05449	1,052191
4	0,93864	1,98493	1,04629	1,044009
5	0,94703	1,99952	1,05249	1,050195
			Average volume	1,049708
			Standard deviation	0,002364
			Factor	1,905291
Test assays	Empty Tube	Tube + volume injected of 1,5 mL (g)	Volume injected of 1,5 mL (g)	Volume injected of 1,5 mL (mL)
1	0,95035	2,4582	1,50785	1,504563
2	0,94535	2,44293	1,49758	1,494315
3	0,9471	2,46165	1,51455	1,511248
4	0,94735	2,4492	1,50185	1,498576
5	0,94208	2,44408	1,502	1,498725
			Average volume	1,501485
			Standard deviation	0,005136
			Factor	1,903406

Appendix C

Bill of Materials

Table C 1 - Bill of materials

3-D Printed	Count	Printing hours	Price / printing hour	Total
Motor End - 20% fill	1	6,65	3,00 €	19,95 €
Carriage - 20% fill	1			
Retainer – Type 1 - 20% fill	2			
Retainer – Type 2 - 20% fill	2			
Clamp – Type 1 - 20% fill	2			
Clamp – Type 2 - 20% fill	2			
Idler End - 20% fill	1			
Motors & Metal			**Price/unit**	
NEMA 17 0.9°/step	1		18,20 €	18,20 €
5mm x 5mm shaft coupling	1		3,60 €	3,60 €
625z ball bearing	2		1,35 €	2,70 €
LM6UU linear bearing	2		1,10 €	2,20 €
M3 x 10mm socket head cap screw	4		0,07 €	0,30 €
M3 x 20mm socket head cap screws	6		0,14 €	0,81 €
M3 x 35mm socket head cap screws	6		0,21 €	1,25 €
M3 hex nut	16		0,05 €	0,79 €
M5 hex nut	1		0,12 €	0,12 €
		Meters	**Price/meter**	
M5 threaded rod 0,2 m	1	0,2	0,49 €	0,10 €
6mm A2 tool steel 0,2 m	2	0,4	1,85 €	0,74 €
			Price/syringe pump:	50,76 €

Final System	Count	Printing hours	Price/unit	
Arduino MEGA 2560 rev3	1		35,00 €	35,00 €
Syringe pump	3		50,76 €	92,43 €
Stepper motor driver	3		8,50 €	25,50 €
3D printed µ-dish lid - 20% fill	1	0,15	3,00 €	0,45 €
			Total:	213,22 €

Appendix D

Arduino Firmware v3

```
/*

   Switch case loops declaration:
   Get Identification: 30
     Returns (asci!) MM-Ard\r\n

   Get Version: 31
     Returns: version number (as ASCII string) \r\n

   Motor extension for AOTFcontroller.ino

    Version changed to version_ = 3 at .ino and Arduino .dll file
(Micro-manager)

    Run Stepper motor: 50pqrsss
    p - pump number
    q - speed (rpm)
    r - diretion
    sss - steps to do 24-bit significant number

    Calibrate Stepper motor: 51xy
    x - pump number
    y - calibration volume
*/
#include <Wire.h>
#include <Arduino.h>
#include "DRV8834.h"
#include <LiquidCrystal_I2C.h>
#include <QueueArray.h>

#define DIR_1 2
#define STEP_1 3
#define M0_1 5
#define M1_1 4

#define DIR_2 6
#define STEP_2 7
#define M0_2 9
#define M1_2 8
```

```
#define DIR_3 10
#define STEP_3 11
#define M0_3 13
#define M1_3 12

#define BUTTON1 44
#define BUTTON2 45
#define BUTTON3 46

#define MOTOR_STEPS 400

DRV8834 stepper1(MOTOR_STEPS, DIR_1, STEP_1, M0_1, M1_1);
DRV8834 stepper2(MOTOR_STEPS, DIR_2, STEP_2, M0_2, M1_2);
DRV8834 stepper3(MOTOR_STEPS, DIR_3, STEP_3, M0_3, M1_3);

int microsteps = 32;
int step_counter = 0;

LiquidCrystal_I2C lcd(0x3F, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);

unsigned int version_ = 3;

const int SEQUENCELENGTH = 12;  // this should be good enough for
everybody;)
byte triggerPattern_[SEQUENCELENGTH] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0};
unsigned int triggerDelay_[SEQUENCELENGTH] = {0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0};
int patternLength_ = 0;
byte repeatPattern_ = 0;
volatile int triggerNr_; // total # of triggers in this run (0-based)
volatile int sequenceNr_; // # of trigger in sequence (0-based)
int skipTriggers_ = 0;  // # of triggers to skip before starting to
generate patterns
byte currentPattern_ = 0;
const unsigned long timeOut_ = 1000;
bool blanking_ = false;
bool blankOnHigh_ = false;
bool triggerMode_ = false;
boolean triggerState_ = false;

void setup() {
  // Higher speeds do not appear to be reliable
  Serial.begin(57600);

  stepper1.setRPM(200);
  stepper1.setMicrostep(microsteps);
  stepper2.setRPM(200);
  stepper2.setMicrostep(microsteps);
  stepper3.setRPM(200);
  stepper3.setMicrostep(microsteps);
  pinMode(BUTTON1, INPUT);
  pinMode(BUTTON2, INPUT);
  pinMode(BUTTON3, INPUT);

  lcd.begin (16, 2);
  lcd.setBacklight(HIGH);

}

void loop() {
```

```
  if (Serial.available() > 0) {
    int inByte = Serial.read();
    switch (inByte) {

      // Gives identification of the device
      case 30:
        Serial.println("MM-Ard");
        break;

      // Returns version string
      case 31:
        Serial.println(version_);
        break;


      /* extension components


      */

      case 50:
        if (waitForSerial(timeOut_)) {
          byte pump_ = Serial.read();
          if (waitForSerial(timeOut_)) {
            byte speed_ = Serial.read();

            if (waitForSerial(timeOut_)) {
              byte direction_ = Serial.read();

              if (waitForSerial(timeOut_)) {
                byte steps_to_do_msb = Serial.read();
                if (waitForSerial(timeOut_)) {
                  byte steps_to_do_isb = Serial.read();
                  if (waitForSerial(timeOut_)) {
                    byte steps_to_do_lsb = Serial.read();

                    lcd.clear();
                    lcd.setCursor(0, 0);
                    lcd.print("Pump  :");
                    lcd.setCursor(5, 0);
                    lcd.print(pump_);
                    lcd.setCursor(7, 0);
                    lcd.print("ON");

                    lcd.setCursor(0, 1);
                    lcd.print("Steps:");
                    lcd.setCursor(6, 1);

                    unsigned long steps_to_do = ((unsigned
long)steps_to_do_msb << 16 ) | ((unsigned long)steps_to_do_isb << 8) |
((unsigned long)steps_to_do_lsb & 0xff);
                    lcd.print(steps_to_do);
                    delay(1000);

                    if (pump_ == 1)
                      stepper1.setRPM((int) speed_);
                    else if (pump_ == 2)
                      stepper2.setRPM((int) speed_);
                    else if (pump_ == 3)
                      stepper3.setRPM((int) speed_);
```

```
                    int direction_to_do = 1;

                    if (direction_ == 1)
                      direction_to_do = -1;

                    while (steps_to_do >= 32000)
                    {
                      if (pump_ == 1)
                        stepper1.move(32000 * direction_to_do);
                      else if (pump_ == 2)
                        stepper2.move(32000 * direction_to_do);
                      else if (pump_ == 3)
                        stepper3.move(32000 * direction_to_do);

                      steps_to_do -= 32000;
                    }
                    if (pump_ == 1)
                      stepper1.move((int)( steps_to_do *
direction_to_do));
                    else if (pump_ == 2)
                      stepper2.move((int)( steps_to_do *
direction_to_do));
                    else if (pump_ == 3)
                      stepper3.move((int)( steps_to_do *
direction_to_do));

                    Serial.write( byte(50));
                    Serial.write( pump_);
                    Serial.write( speed_);
                    Serial.write( direction_);
                    Serial.write( steps_to_do_msb);
                    Serial.write( steps_to_do_isb);
                    Serial.write( steps_to_do_lsb);
                  }
                }
              }
            }
          }
        }

        break;

      case 51:
        if (waitForSerial(timeOut_)) {
          byte pump_ = Serial.read();

          if (waitForSerial(timeOut_)) {
            byte volume_ = Serial.read();
            lcd.clear();
            lcd.setCursor(0, 0);
            lcd.print("Set Pump   Sy-");
            lcd.setCursor(9, 0);
            lcd.print(pump_);
            lcd.setCursor(0, 1);
            lcd.print("ringe to:");
            lcd.setCursor(9, 1);
            lcd.print(volume_);
            lcd.setCursor(11, 1);
            lcd.print(" mL");
```

```
            while (digitalRead(BUTTON3) == LOW) {
              if (digitalRead(BUTTON1) == HIGH)
              {
                if (pump_ == 1)
                  stepper1.move(100 * microsteps);
                else if (pump_ == 2)
                  stepper2.move(100 * microsteps);
                else if (pump_ == 3)
                  stepper3.move(100 * microsteps);
              }
              if (digitalRead(BUTTON2) == HIGH)
              {
                if (pump_ == 1)
                  stepper1.move(-100 * microsteps);
                else if (pump_ == 2)
                  stepper2.move(-100 * microsteps);
                else if (pump_ == 3)
                  stepper3.move(-100 * microsteps);
              }
            }



            lcd.clear();
            lcd.setCursor(0, 0);
            lcd.print("Calibration done");
            delay(1000);
            lcd.clear();

            Serial.write( byte(51));
            Serial.write( pump_);
            Serial.write( volume_);
          }
        }

        break;


    }
  }
}


bool waitForSerial(unsigned long timeOut)
{
  unsigned long startTime = millis();
  while (Serial.available() == 0 && (millis() - startTime < timeOut) )
{}
  if (Serial.available() > 0)
    return true;
  return false;
}
```